

Disaster tweet text and image analysis using deep learning approaches

by

Xukun Li

BS, Xiamen University, 2014

MS, Kansas State University, 2017

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2020

Abstract

Fast analysis of damage information after a disaster can inform responders and aid agencies, accelerate real-time response, and guide the allocation of resources. Once the details of damage information have been collected by a response center, the rescue resources can be assigned more efficiently according to the needs of different areas. The challenge of the information collection is that the traditional communication lines can be damaged or unavailable in the beginning of a disaster. With the fast growth of the social media platforms, the situational awareness and damage data can be collected, as affected people will post information on social media during a disaster. There are several challenges to analyzing the social media disaster data. One challenge is posed by the nature of social media disaster data, which is generally large, but noisy, and comes in various formats, such as text or images. This challenge can be addressed by using deep learning approaches, which have achieved good performance on image processing and natural language processing. Another challenge posed by disaster related social media data is that it needs to be analyzed in real-time because of the urgent need for damage and situational awareness information. It is not feasible to learn supervised classifiers in the beginning of a disaster given the lack of labeled data from the disaster of interest. Domain adaptation can be applied to address this challenge. Using domain adaptation, we can adapt a model learned from pre-labeled data from a prior source disaster to the current on-going disaster. In this dissertation, I propose deep learning approaches to analyze disaster related tweet text and image data. Firstly, domain adaptation approaches are proposed to identify informative images or text, respectively. Secondly, a multimodal approach is proposed to further improve the performance by utilizing the information from both images and text. Thirdly, an approach for localizing and qualifying damage in the informative images is proposed. Experimental results show that the proposed approaches

can efficiently identify informative tweets or images. Furthermore, the type and the area of damage can be localized effectively.

Disaster tweet text and image analysis using deep learning approaches

by

Xukun Li

BS, Xiamen University, 2014

MS, Kansas State University, 2017

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2020

Approved by:

Major Professor
Doina Caragea

Copyright

© Xukun Li 2020.

Abstract

Fast analysis of damage information after a disaster can inform responders and aid agencies, accelerate real-time response, and guide the allocation of resources. Once the details of damage information have been collected by a response center, the rescue resources can be assigned more efficiently according to the needs of different areas. The challenge of the information collection is that the traditional communication lines can be damaged or unavailable in the beginning of a disaster. With the fast growth of the social media platforms, the situational awareness and damage data can be collected, as affected people will post information on social media during a disaster. There are several challenges to analyzing the social media disaster data. One challenge is posed by the nature of social media disaster data, which is generally large, but noisy, and comes in various formats, such as text or images. This challenge can be addressed by using deep learning approaches, which have achieved good performance on image processing and natural language processing. Another challenge posed by disaster related social media data is that it needs to be analyzed in real-time because of the urgent need for damage and situational awareness information. It is not feasible to learn supervised classifiers in the beginning of a disaster given the lack of labeled data from the disaster of interest. Domain adaptation can be applied to address this challenge. Using domain adaptation, we can adapt a model learned from pre-labeled data from a prior source disaster to the current on-going disaster. In this dissertation, I propose deep learning approaches to analyze disaster related tweet text and image data. Firstly, domain adaptation approaches are proposed to identify informative images or text, respectively. Secondly, a multimodal approach is proposed to further improve the performance by utilizing the information from both images and text. Thirdly, an approach for localizing and qualifying damage in the informative images is proposed. Experimental results show that the proposed approaches

can efficiently identify informative tweets or images. Furthermore, the type and the area of damage can be localized effectively.

Table of Contents

List of Figures	xii
List of Tables	xv
Acknowledgements	xviii
1 Introduction	1
1.1 Motivation	1
1.2 Disaster Data Analysis	2
1.3 Deep Learning	4
1.4 Multimodal Approach	5
1.5 Domain Adaptation	5
1.6 Outline and Contributions	6
2 Identifying Disaster Damage Images Using a Domain Adaptation Approach	9
2.1 Introduction	9
2.2 Related Work	12
2.3 Background and Approach	13
2.3.1 Adapted VGG-19 Architecture	13
2.3.2 Domain-Adversarial Neural Networks	15
2.4 Experimental Setup	18
2.4.1 Dataset	19
2.4.2 Setup	19
2.4.3 Baselines	20

2.4.4	Implementation Details	20
2.5	Experimental Results	20
2.5.1	Discussion of the Results	22
2.5.2	Estimating the Distance between Source and Target Domains before and after Adaptation	23
2.5.3	Visual Distribution Analysis	26
2.6	Conclusions and Discussion	27
3	Domain Adaptation with Reconstruction on Disaster Tweets	29
3.1	Introduction	29
3.2	Approach	31
3.2.1	Domain Adaptation	31
3.2.2	Recurrent Neural Networks	32
3.2.3	Seq2Seq Autoencoder	32
3.2.4	Domain Adaptation with Reconstruction	33
3.3	Experimental Setup	34
3.3.1	Implementation Details	34
3.3.2	Datasets	34
3.3.3	Research Questions	35
3.4	Experimental Results	36
3.4.1	Results	36
3.4.2	Discussion	37
3.5	Conclusions	38
4	Improving Disaster related Tweet Classification with Using a Multimodal Deep Learning Approach	39
4.1	Introduction	39
4.2	Related Work	41

4.3	Approach	42
4.3.1	RNN	44
4.3.2	CNN	44
4.3.3	Multi-Modal Model	45
4.4	Experimental Results	46
4.4.1	Dataset	46
4.4.2	Experimental Setup	47
4.4.3	Evaluation Metric	48
4.4.4	Experimental Results	49
4.5	Conclusions	53
5	Localizing and Quantifying Damage in Social Media Images	55
5.1	Introduction	55
5.2	Proposed Approach	58
5.2.1	Convolutional Neural Networks	58
5.2.2	Damage Detection Map	60
5.2.3	Measuring the damage severity	62
5.3	Experimental Results I	64
5.3.1	Experimental Setting	64
5.3.2	Damage Detection Map Evaluation	65
5.3.3	Damage Assessment Value Evaluation	67
5.3.4	Classification using DAV values	69
5.4	Experimental Results II	70
5.4.1	Experimental Setup	70
5.4.2	Experimental Results and Discussion	76
5.5	Related Work	83
5.6	Conclusion	84

6	Conclusions and Future Work	86
A	Optimally of the Adversarial Training	104
A.1	Global Optimally of $P_{T(X_s)} = P_{T(X_t)}$	104

List of Figures

1.1	Hierarchical Tweet Classification	3
2.1	Overview of the Model Architecture.	14
2.2	Examples of images that are correctly classified or miss-classified by different networks, when Ruby Typhoon is used as target disaster and Ecuador Earthquake is used as source disaster. (Top) Images that are classified correctly as <i>damage</i> by DANN-SL-TU and VGG-19-TL, and miss-classified as <i>no-damage</i> by VGG-19-SL. (Middle) Images that are miss-classified as <i>damage</i> by DANN-SL-TU and VGG-19-TL, while VGG-19-SL correctly classifies them as <i>no-damage</i> . (Bottom) The first two images on the left are miss-classified by both DANN-SL-TU and VGG-19-SL, while VGG-19-TL correctly classifies image (h) as <i>damage</i> and image (i) as <i>no-damage</i> . The last two images on the right have label <i>no-damage</i> and are miss-classified by all networks as <i>damage</i> .	24
2.3	Two dimensional representation of source (blue) and target (red) using the VGG-19-SL representation of the images (before adaptation), and the DANN-SL-TU transformed representation of the images (after adaptation), for two pairs of disasters. The pair on the left consists of two dissimilar disasters, Ecuador Earthquake (source, blue) and Ruby Typhoon (target, red). The pair on the right consists of two similar disasters, Ecuador Earthquake (source, blue) and Nepal Earthquake (target, red).The two representations are reduced to 2 dimensions using the t-SNE [1] technique.	26
3.1	Architecture of the proposed approach	32

4.1	Architecture of proposed model	43
5.1	Overview of the proposed approach.	58
5.2	Examples of images in our dataset, and their corresponding DDM heatmaps and DAV scores. The first row shows examples of images in the <i>damage</i> class, followed by their corresponding DDM heatmaps and the DAV scores. Similarly, the third row shows examples of images in the <i>no damage</i> class, followed by their corresponding DDM heatmaps and DAV scores.	63
5.3	Damage Detection Map (DDM) versus human annotations for images 1, 5 and 7 in Table 5.2. (a) Original Google image; (b) DDM damage heatmap; (c) Binary representation of the DDM computed with a threshold whose value is 20% of the max value in the DDM heatmap; (d) Damage marked by Annotator A; (e) Damage marked by Annotator B.	68
5.4	Examples of images mislabeled in the disaster dataset. The original image label and the DAV score are shown.	69
5.5	Smoothed density curves for DAV values. The <i>no damage</i> class is shown in blue and the <i>damage</i> class is shown in red. The graphs are based on the test set of each disaster dataset.	69
5.6	Sample images together with their corresponding damage localization annotation for <i>building-damage</i> , <i>bridge-damage</i> and <i>road-damage</i> , respectively. . .	73
5.7	Sample images from the Building Damage Severity dataset, together with their severity annotations	74

5.8	Visualization of Damage Detection Maps for a building image (top two rows), a bridge image (middle two rows) and a road image (last two rows). The first row corresponding to an image shows the original image and the DDM maps produced with a six-class (6c), four-class (4c) and two-class (2c) VGG19 model, respectively. The Grad-CAM (GCAM) is used with all models. For each model, the IoU value of the image is also shown. The second row corresponding to an image shows the manual damage annotation of the image, together with the binary maps obtained from the DDM heatmaps (the binary maps are used to calculate the IoU values).	75
5.9	Heatmaps (top row) and the corresponding binary maps (bottom row) for the sample images from the Building Damage Severity dataset, together with their severity annotations (shown below the top images) and the DAV scores (shown below the bottom images). The DAV scores properly reflect the severity of the damage.	80
5.10	Examples of images with DDM heatmaps that lead to high/low IoU values. The first two rows show images with high IoU values, while the last two rows show images with low IoU values.	81

List of Tables

2.1	Disaster class distribution, together with the total number of labeled images in each disaster.	19
2.2	Classification results: average accuracy (Acc.), precision (Prec.), recall (Rec.) and F1-measure (F1) for the damage class (average is taken over three runs), together with the corresponding standard deviation (in parentheses), for DANN-SL-TU (which uses source labeled data and target unlabeled data), VGG-19-SL (adapted using source labeled data), and VGG-19-TL (adapted using target labeled data). Highlighted in bold font are results that are statistically different when comparing VGG-19-SL and DANN-SL-TU using a paired t-test with $p \leq 0.05$	21
2.3	Proxy \mathcal{A} -distance between different domains. In the ‘After Adaptation’ case, $(A \rightarrow B)$ means that the model was trained with A as source and B as target, while $(B \rightarrow A)$ means that the model was trained with B as source and A as target.	25
3.1	Statistics of the dataset published in [2]	35
3.2	Statistics of the the CrisisMMD dataset [3]	35
3.3	Cross-domain and in-domain results.	37
3.4	Cross-domain and in-domain (e.g., $D0 \rightarrow D0$) Accuracy, AUC and F1 for CrisisMMD. Results based on 5 independent runs.	38
4.1	Tweet text and image examples	40
4.2	Statistics of the the dataset CrisisMMD	47
4.3	Hurricane Irma text/image distribution	47

4.4	Experimental results for the first scenario (image/text matched label results). D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods	50
4.5	Text label results. D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods	51
4.6	Image label results. D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods	51
4.7	Text Classification Examples: the multimodal model helps improve the classification	53
5.1	Social media image dataset consisting of images from four disaster events. Images are labeled as <i>Severe</i> , <i>Mild</i> , or <i>None</i> . The number of images in each class, and the total number of images for each disaster are shown.	64
5.2	IOU for 10 Google images. Annotation A and Annotation B are provided independently by two annotators. The image heatmaps were transformed to a binary representation by using a threshold equal to 20% of the max value in the DDM	67
5.3	Classification accuracy for each disaster dataset. The first row shows the average accuracy of our simple DAV-based classifiers (over 5 independent runs), together with standard deviation. The second row shows the average accuracy of the three-class CNN models. The third row shows the results published in [4], which used the same datasets.	70
5.4	Statistics about the Google damage dataset, which contains images in the following categories: <i>bridge-damage</i> , <i>no-bridge-damage</i> , <i>building-damage</i> , <i>no-building-damage</i> , <i>road-damage</i> , and <i>no-road-damage</i>	72

5.5	Statistics about the Building Damage Severity dataset, which contains building images in the following categories: <i>no-damage</i> (0), <i>slight-damage</i> (0.25), <i>moderate-damage</i> (0.5), <i>heavy-damage</i> (0.75), and <i>total-destruction</i> (1)	72
5.6	Experimental results on the Google test dataset	79

Acknowledgments

I would like to express my deepest gratitude to my advisor Dr. Doina Caragea, for giving me the wonderful opportunity to participate in the research of disaster image processing and natural language processing. This dissertation would not have been possible without her help. What she taught me is not only the research skills but also how to come up with research ideas. I am very thankful to my committee members Dr. Torben Amtoft, Dr. Dan Andresen, and Dr. Caterina Scoglio, for their participation and the valuable suggestions during my research.

Chapter 1

Introduction

1.1 Motivation

It is important to achieve fast response during a disaster to rescue lives and protect properties. One challenge is that the disaster response teams may only have limited information about the damage situation because the communication systems may become damaged or unavailable at the beginning of a disaster. Nowadays, with the fast growth of social media platforms such as Twitter and Facebook, people on the ground usually post a variety of content in the form of textual messages and images during a disaster. It is useful to get information from the social media platforms to gain situational awareness, analyze the severity of the damage and inform the rescue operations [5, 6, 7]. For example, if they can get information about injured people in some area, then they can send medical teams to that area or adjacent areas where medical services may be needed. Similarly, if they have information about damaged roads or bridges, they can send engineers to fix the roads and bridges and enable the traffic in the affected areas. This kind of information potentially provided by eyewitnesses of the disaster on social media can help optimize the rescue process, especially when the resources are limited at the beginning of the disaster.

One challenge faced by social media disaster data analysis is that there is a large amount of data posted online. We need to identify the text or images that contain useful information,

and then extract that information. Another challenge is that the data posted on the social media platform can take multiple forms, such as text or images. Text or images have a complicated intricate structure that may not be effectively learned from traditional machine learning algorithms. The traditional machine learning models which have a simple model structure may not learn a good text or image representation as the relationship between words in a sentence or pixels in an image can be very complex [8, 9]. Furthermore, given that the tweet data usually contains both text and image, the models used in disaster data analysis need to be able to consume the information from both text and images. A third challenge is posed by the need to analyze the data fast, ideally in real-time. The faster the relevant information is extracted, the more informed and better the rescue process, because the first few days are the ‘golden time’ to rescue people. This is not easy to do using supervised learning, which requires time-consuming data labeling. This dissertation addresses the above challenges. We discuss how to filter informative data and what information we can extract in Section 1.2 to address the first challenge; deep learning and multimodal approaches for analyzing text and images are discussed in Sections 1.3 and 1.4, respectively; one possible solution for achieving fast analysis, called domain adaptation, is discussed in Section 1.5; the outline and contributions of this dissertation are presented in Section 1.6.

1.2 Disaster Data Analysis

When a disaster occurs, an important task for response organizations is to extract useful information from “chaos”. Lots of tweets and images are posted on social media platforms, but many of them are not useful, as these data may not include the information needed by response teams. Thus, the first step in disaster data analysis is to identify informative tweets or images. It is straightforward to search by keyword, for example, search with the keyword “hurricane Irma”. But the search results are still noisy and may miss some useful information if users posting informative tweets do not include the keywords searched. For example, the tweet: “With Harvey’s havoc fresh on their minds, Floridians brace for Hurricane Irma” includes the keywords “Hurricane Irma” but there is no useful information for the response

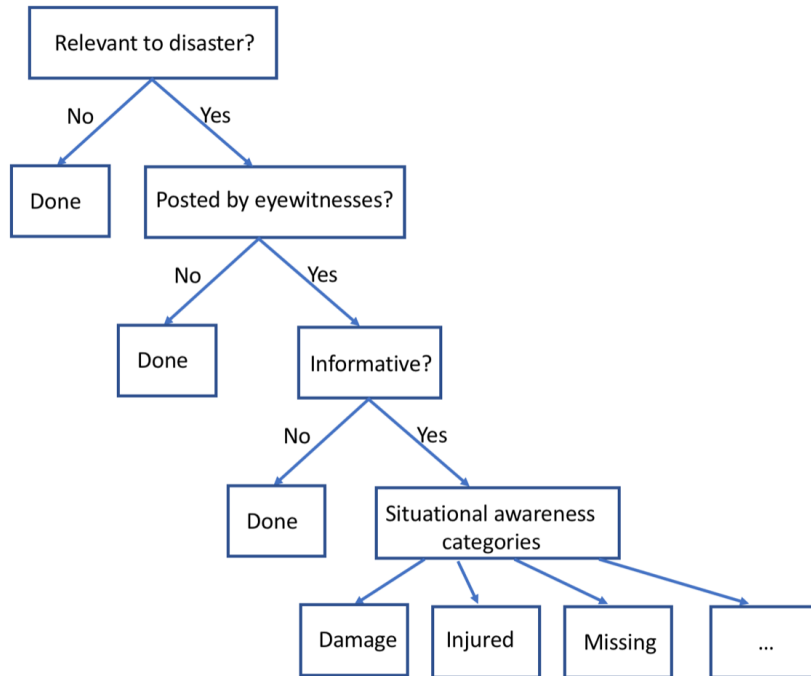


Figure 1.1: Hierarchical Tweet Classification

teams. In contrast, this tweet: “Merritt Island community hit hard by likely Hurricane Irma-spawned tornado” can help the response organizations to allocate rescue resources for people in that area. We need machine learning algorithms to identify informative tweets or images instead of simply performing query-based filtering. When data and labels are available, machine learning models can be learned to classify each tweet/image into categories, such as informative/not informative.

Other useful information can be extracted from tweets or images after filtering relevant data. For example, we can identify eyewitness tweets, as those are the most useful to the response teams, and in particular situational awareness tweets posted by eyewitnesses. Furthermore, different types of damage and information about injured or missing people can be identified. As shown in Figure 1.1, the information extraction can be conducted in a hierarchical manner, where a classifier can be built for each task in a top-down fashion. Thus, instead of having one model to predict all types of useful information, several classification models can be trained independently, with one model being used on the output of the parent model. In terms of learning algorithms, the same algorithm can be used for multiple tasks.

Several studies have applied classification models to identify relevant/informative disaster tweets [10, 11, 12, 13], eyewitness tweets [14, 15], situational awareness tweets [16, 17], and damage severity in disaster images [18], among others.

1.3 Deep Learning

A tremendous amount of disaster-related data are in the format of text or images. Traditional machine learning models may not be good at extracting features from such data or may require extensive human effort to extract the features. Nowadays, with the fast growth of deep learning, lots of methods have been published for analyzing images and text. The state-of-art methods for image and text classification are convolutional neural networks (CNN) and recurrent neural networks (RNN).

Convolutional neural networks (CNN) [19] have achieved impressive results in image classification and text classification. Many network architectures have been proposed for image classification in recent years, including AlexNet [20], VGG-19 [21], ResNet [22]. RNN [23] can be used in time-series data, for example, text data. It has been applied to text classification and achieved promising results [24]. RNN models include Long Short Term Memory (LSTM) networks [25] or Gated Recurrent Units (GRU) [26], which capture short and long terms dependencies in text, to further improve the classification performance.

In disaster analysis, these models can be applied to disaster text and images posted on the social media platforms to provide information to response organizations. Some studies have applied deep learning approaches to disaster data. It has been shown that the CNN model can achieve better results than the Naive Bayes model for identifying informative tweets during disasters [27]. Nguyen et al. [18] has applied the CNN model to assess the damage severity in disaster images.

1.4 Multimodal Approach

Given that the data in the social media platforms usually contain both text and images, it is beneficial to combine the information from text and images into one model to improve the overall classification performance. The first challenge of multimodal learning is how to combine the text and images into one model. One possible way to combine the text and images is to train two models for text and images independently, and average the output as the final classification decision. Another way is to concatenate the output of text and image models and co-training the two models by gradient descent, which is often used in the optimization of deep learning models. Some multimodal approaches have been published, and the two possible solutions outlined above have been explored in the deep learning literature [28, 29, 30]. However, the application of multimodal approaches in disaster data analysis is still in its infancy. Furthermore, prior studies do not address scenarios where the text and the corresponding image have different labels, which can happen frequently in disaster data analysis. Users may post a tweet with an unrelated image or post image with an unrelated text. Most of the approaches published will output one classification decision that cannot be applied to the data when the text and the image have different labels. In this dissertation, we propose a deep learning based multimodal approach for disaster data analysis, and experiment with different scenarios to address the challenge that text and image may have a different label.

1.5 Domain Adaptation

Another challenge for the disaster tweet classification task is posed by the need for fast response. Traditional supervised learning requires labeled data, which is generally expensive. It makes the classification task difficult while the label for the current disaster is unavailable. One possible solution is to use Domain Adaptation [31]. Domain adaptation aims to learn a model from a source data (previous disaster dataset which was pre-labeled) and possibly target unlabeled data (currently ongoing disaster). Subsequently, the model is used

to identify useful tweets in the target data. Thus, domain adaptation eliminates the need for labeled target data, potentially allowing for improvements in terms of response time. Usually, domain adaptation will combine supervised learning (used for source domain) and unsupervised or self-supervised learning (used for target domain).

There are several popular deep learning based domain adaptation approaches. One approach is to make the representation for different domains similar. Domain adversarial neural network (DANN) model proposed by Ganin et al. [32] is in this category. This model is based on adversarial training which can generate images with similar distribution from a set of target images. The difference between two domains (source and target) can be reduced by DANN, then the classifier trained on source can be used on the target. There are other adversarial based methods such as Domain Adversarial Residual Transfer Networks (DART) [33], Domain Flow (DLOW) [34] and Cycle Consistent Adversarial (CyCADA) [35]. Another type of approach is to conduct semi-supervised learning by reconstructing the target domain [36]. The target domain will be reconstructed by an encoder and decoder. Then the encoder will learn the information from the target domain and the representation obtained can be used to learn a classifier from the source domain.

Domain adaptation is valuable in disaster data analysis. At the beginning of a disaster, response organizations need to react fast, which makes it almost impossible to label the target data given that the process of labeling data is time-consuming. The domain adaptation approach is better suited for this situation compared to supervised learning which requires a labeled dataset for the ongoing disaster. Several studies have been published to apply the domain adaptation approach in disaster data analysis [10, 11, 12]. However, deep learning based domain adaptation approaches, which might achieve better performance on disaster text and images, are still under exploration.

1.6 Outline and Contributions

In this dissertation, I propose several approaches to analyze disaster data. First, domain adaptation approaches are proposed to identify informative images and text, in Chapter

2 and Chapter 3, respectively. The identification performance is further improved by a multimodal approach presented in Chapter 4. Finally, an approach is proposed to localize and qualify damage in the disaster images Chapter 5. The major published contributions of this dissertation include:

- Chapter 2: *Identifying Disaster Damage Images Using a Domain Adaptation Approach* [37]

In this chapter, we propose a domain adaptation approach, which is based on domain adversarial training, to identify disaster damage images. This approach can reduce the difference between the source and target domains. Thus, the damage images from the target domain can be identified by utilizing pre-labeled images from the source disaster. The damage images identified from the target domain can be used to perform fast damage assessment.

- Chapter 3: *Domain Adaptation with Reconstruction on Disaster Tweets* [38]

In this chapter, we propose a domain adaptation approach for classifying informative tweets. The method used was adapted from Domain Reconstruction Neural Networks. We trained an autoencoder model (seq2seq model) to reconstruct the target data and an RNN model to classify source tweets. The autoencoder and RNN models share the feature extraction layer. The information from the target domain is extracted from the autoencoder, and the classifier trained on the source domain is applied to label data from the target domain. The proposed method was evaluated on two publicly available datasets.

- Chapter 4: *Improving Disaster-related Tweets Classification with a Multimodal Approach* [39]

In this chapter, we propose an approach which utilizes both text and images. This multimodal approach combines a CNN model, which extract feature from images, and an RNN model, which learns representations for text. A fully-connected layer is applied to the concatenated representations to predict the class label. The proposed

multimodal approach can be used to improve the text classification performance, by adding the image information.

- Chapter 5: *Localizing and Quantifying Damage in Social Media Images* [40, 41]

In this chapter, we propose an approach adapted from Grad-CAM [42], called Damage Detection Map (DDM), to generate a smooth damage heatmap for an image. Based on the damage heatmap, we also find a Damage Assessment Value (DAV), to quantify the severity of the damage. Furthermore, we compare Grad-CAM and Grad-CAM++ [43] in the context of damage localization and quantification. We also explore the proposed approach on different types of infrastructure damage (including building damage, bridge damage, and road damage).

Chapter 2

Identifying Disaster Damage Images Using a Domain Adaptation Approach

2.1 Introduction

The increased popularity of social media websites has transformed the way in which affected populations communicate with response organizations during and following major disasters. Victims of several recent hurricanes, including Hurricane Harvey, Hurricane Irma and Hurricane Florence, have turned to social media to request help and assistance, as emergency hotlines were sometimes unreachable due to the high volume of calls [44, 45, 46]. According to Rhodan [44], a woman and two children were rescued during Hurricane Harvey after the woman posted a desperate message for help on Twitter: ‘I have 2 children with me and tge [sic] water is swallowing us up. Please send help.’ Similarly, an image posted on Twitter, which showed ‘elderly residents sitting in greenish flood water,’ raised awareness of a nursing home situation and resulted in urgent help being sent to the nursing home [44].

While social media can help increase situational awareness, inform rescue operations, and save lives, its value is highly unexploited by response teams, in part due to the lack of tools that can help filter actionable information from the big data posted during disasters. A recent study [47] reported that during Hurricane Harvey, FEMA missed 46% of the critical

damage information posted by affected individuals on Twitter, and thus many areas heavily impacted by the hurricane were missed from the original damage estimates provided by FEMA. Similarly, Kryvasheyev et al. [48] used Hurricane Sandy tweets, and Enenkel et al. [49] used Hurricanes Harvey and Irma filtered, geo-located tweets to show that rapid early damage assessment can be facilitated by social media. Among others, the above-mentioned studies [48, 47, 49] suggest that tools that can identify critical social media information in real-time are greatly needed, and should be incorporated into the operational decision-making pipelines of response organizations.

To meet these requirements, many studies have focused on the design of machine learning tools to identify relevant, informative, actionable situational awareness information in social media [50, 51, 52, 53]. More recently, deep learning approaches have also been proposed in the context of identifying information useful for disaster response on social media [54, 55, 27, 56, 57], and have been generally shown to produce better results than the traditional supervised learning approaches. Research on classifying and retrieving useful information from disaster images is still in its early stage, although recent studies have suggested that useful social awareness information can be found in social media images [58, 59, 60]. To advance the state-of-the-art in this area, Alam et al. [3] and Mouzannar et al. [61] recently published multi-modal datasets, consisting of both tweet text and images. Furthermore, Mouzannar et al. [61] developed a deep learning approach to identify damage images in their dataset. Related to this, Nguyen et al. [62] proposed to classify disaster images according to damage severity using fine-tuned convolutional neural networks, and Li et al. [40] proposed a method based on class activation mapping (CAM) to localize and quantify damage in social media images posted during disasters.

While significant contributions have been made in terms of developing traditional and deep learning models to identify useful situational awareness information in social media, most of the existing approaches are supervised learning approaches, which require labeled data to train accurate models. It is unrealistic to assume that labeled data is readily available in the early hours of a disaster, when help may be most needed. Domain adaptation or transfer learning approaches [63], which make use of labeled data from a prior disaster

(called *source*) and unlabeled data from the emergent disaster (called *target*), have been shown to lead to better results as compared to supervised classifiers learned from a prior disaster, when used for target tweet classification [64, 65]. Domain adaptation has been used extensively for image classification using deep neural networks [66], including social media image classification by fine-tuning a pre-trained deep neural network, such as VGG-16 or VGG-19 [21], as in [62, 40]. However, domain adaptation from a source disaster to a target disaster has not been explored in the context of classifying social media images posted during and shortly after a disaster. Thus, the goal of this study is to gain insights into the usefulness of transferring knowledge from a source disaster to a target disaster, when classifying images according to damage severity.

We propose to use an approach called Domain-Adversarial Neural Network (DANN) introduced by Ganin et al. [32]. The DANN approach is based on domain-adversarial training, which reduces the shift between the source and target domains, by making the source and target feature representations indiscriminate, while the source feature representation is discriminative for the source classification task. Experimental results on images from several source-target disaster pairs have shown that the DANN approach, which uses source labeled data and unlabeled target data, can improve the classification accuracy obtained with a deep neural network trained only from source labeled data, especially if the two disasters are of different type (e.g., a hurricane and an earthquake).

The contributions of the paper are as follows:

- We have adapted the DANN model by combining it with VGG-19 to benefit from the VGG-19 extensive training.
- We have performed extensive experiments on several pairs of disasters, and studied the ability of the adapted model to transfer information about image damage from a source disaster to a target disaster.
- We have performed visualization of the original and transformed representations of the source versus target images to gain insights into the results of the model.

The rest of this paper is organized as follows: We discuss “RELATED WORK” in the section following the “INTRODUCTION”. Details of the domain adversarial neural network are presented in the “BACKGROUND AND APPROACH” section. The experimental setup and results are described in “EXPERIMENTAL SETUP” and “EXPERIMENTAL RESULTS” sections, respectively. Finally, we conclude the paper, discuss the limitations of our work, and also several ideas for future work.

2.2 Related Work

Domain adaptation on image data has received much attention in recent years [67]. In particular, approaches based on Convolutional Neural Network (CNN) have been very successful, for example, Siamese networks [68] such as the one proposed in [69]. Approaches based on reconstruction have also become popular. For example, Ghifary et al. [36] applied image reconstruction on target unlabeled data by using a CNN autoencoder. Together with the reconstruction model, a supervised model was also trained on the source labeled data. The resulting model was used to retrieve information from target data. Ganin et al. [32] applied adversarial training to adapt data from a source domain to a target domain. The adversarial training was originally used to generate synthetic images [70]. In domain adaptation, the adversarial training learns a data transformation which makes the source and target data to have similar distributions. Then, the classifier trained on the transformed source data can be used on the target data.

In disaster response, domain adaptation is a desirable approach, as it enables fast reaction when a disaster occurs. Several studies have applied domain adaptation approaches on disaster related text data. For example, Li et al. [64] used the iterative Self-Training [71] strategy, with Naive Bayes as a base classifier, to perform domain adaptation for identifying tweets related to a disaster. Domain adversarial training has been applied to disaster tweet classification [2]. Compared to the research on disaster tweet classification, research on disaster image classification and analysis is more limited as of now. Bica et al. [58] performed a visual analysis of two 2015 Nepal Earthquakes, and found a positive correlation between

damage severity and the number of geotagged images posted by eyewitnesses of the earthquakes. Lagerstrom et al. [59] studied the ability of machine learning approaches to identify Twitter images posted during a bush fire emergency situation, and concluded that the fire images can be identified with high accuracy.

More recently, Alam et al. [3] and Mouzannar et al. [61] recently published text and image multi-modal datasets, which have the potential to advance the state-of-the-art in disaster image analysis. Furthermore, Mouzannar et al. [61] developed a deep learning approach to classify images based on damage. Similarly, Nguyen et al. [62] used an approach based on CNNs to classify disaster images according to damage severity. Given that damage is a concept quantifiable on a continuous scale, Li et al. [40] used the CAM approach to localize and quantify damage in disaster images. While these studies represent important steps towards disaster images analysis in real time, they assume target labeled data is available. Domain adaptation approaches, while greatly needed, have not been studied until now.

2.3 Background and Approach

In this section, we introduce some background on image classification, and describe the approach used to perform domain adaptation from a source disaster to a target disaster. Figure 2.1 shows an overview of the model used, which has the VGG-19 network [21] as its backbone, and uses the domain-adversarial training [32] to reduce the domain shift.

2.3.1 Adapted VGG-19 Architecture

Convolutional neural networks (CNN) have achieved impressive results in image analysis and computer vision. Many network architectures have been proposed for image classification in recent years, including AlexNet [20], VGG-19 [21], ResNet [22]. ResNet and VGG-19 models have won the ImageNet competition¹, which proves that these models have good performance on image classification. We choose VGG-19 as the backbone for our model

¹<http://image-net.org/challenges/LSVRC/>

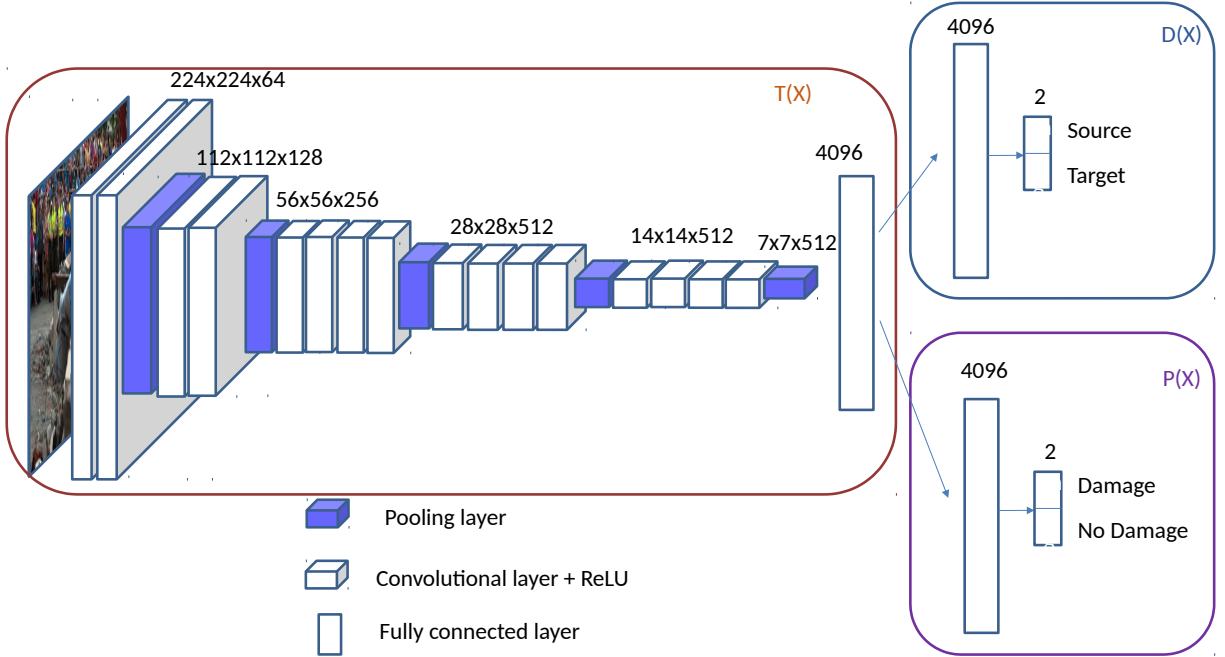


Figure 2.1: Overview of the Model Architecture.

given that this network is relatively simpler than ResNet.

As can be seen in Figure 2.1, VGG-19 consists of 16 convolutional layers and 3 fully connected layers. A convolutional layer can be seen as a feature extractor, as it extracts image fragments (e.g., edge) that are important with respect to the classification task, while ignoring noisy fragments which do not contribute to classification. Each convolutional layer is accompanied by a non-linear ReLU activation. The 5 max pooling layers shown in the figure are used to achieve dimensionality reduction and further filter out noise. The standard VGG-19 has two fully connected layers with dimension 4096, and one softmax layer with dimension 1000. The softmax layer has dimension 1000 because VGG-19 was originally trained for a 1000-class object classification task. However, we aim to adapt VGG-19 to classify disaster images in two classes: *damage* or *no-damage*, and thus, we have changed the dimension of the softmax layer from 1000 dimensions to 2 dimensions. The VGG-19 model includes more than 130 million parameters that need to be learned. It is unfeasible to learn all those parameters from scratch. Instead, we have used the pre-trained VGG-19 model to initialize the parameter values in our network for all, but the last fully connected

layer (as that layer has a different dimension), and subsequently fine-tuned all parameters of the model based on our disaster data. Given that the pre-trained VGG-19 model is learned from millions of images, it is known to generalize well to other sets of images in terms of informative features that it extracts. Thus, we can fine-tune it to our specific classification task with a relatively small number of images.

2.3.2 Domain-Adversarial Neural Networks

Domain Adaptation is a machine learning setting, where data from a source domain, X_s , is used to predict a target domain, X_t , under the assumption that the source and target domains have different distributions (a.k.a., domain shift), but share some similar patterns. In unsupervised domain adaptation, the labels of the source domain data, denoted as y_s , are available, while the labels of the target domain data, y_t , are not available. The task is to learn a classifier for the target data, using the labeled source data and the unlabeled target data. A standard approach to domain adaptation is to transform the source and target data representation (a.k.a., feature adaptation), so that the source and target distributions become indistinguishable [63]. More precisely, one needs to identify a transfer function, $T(X)$, to transform the original source, X_s , and target, X_t , domain data into $T(X_s)$ and $T(X_t)$ data which have similar distributions. Intuitively, a classifier learned on $T(X_s)$ with labels y_s should presumably have a good predictive performance on $T(X_t)$. Thus, the domain adaptation problem is reduced to finding the transfer function $T(X)$. Inspired by the generative adversarial network (GAN) proposed by Goodfellow et al. [70], Ganin et al. [32] designed a Domain-Adversarial Neural Network (DANN), which includes a component that explicitly aims to reduce the shift between a source and a target.

We have adapted the DANN model proposed by Ganin et al. [32] by combining its adversarial training with the VGG-19 network, to take advantage of the extensive training of VGG-19. The architecture of the adapted DANN network, is shown in Figure 2.1. As can be seen in Figure 2.1, our model includes three components: 1) a transfer network, denoted by T , which enables the transfer of information from the source to the target, by reducing

the shift between the two domains (in our case, between the two disasters); 2) a domain discriminative network, denoted by D , which identifies if an input image is from the source or from the target domain; 3) a prediction network, denoted by P , which is used to predict if an image is in the *damage* category or in the *no-damage* category. The input for the transfer network, T , consists of images from either source or target domain. After applying the transformation T to the source or target images, the transformed images are classified as *source* or *target* by the discriminative network D , and classified as *damage* or *no-damage* by the predictor network P . Informally, the idea of the adversarial training is to train a good domain discriminant to separate the source and target domains, and at the same time, train a transfer network to make the two domain look so similar, that even a good discriminant network can not separate them. The predictor network, P , is trained on the transformed source data and then used to classify the transformed target data.

As can be seen in Figure 2.1, the transfer network, $T(X)$, consists of the first 16 convolutional/pooling layers of VGG-19, together with the first fully connected layer in VGG-19. The domain discriminative network, $D(X)$, and prediction network, $P(X)$, have similar architectures, which are equivalent to the last two fully connected layers of VGG-19. Specifically, $D(X)$ has one fully connected layer with dimension 4096, and one fully connected softmax layer with two dimensions, corresponding to the source (S) and the target (T) domains, respectively. Similarly, $P(X)$ has one fully connected layer with dimension 4096, and one fully connected softmax layer with two dimensions, corresponding to *damage* and *no-damage* categories, respectively.

DANN Model Training

The DANN is trained by minimizing the total loss of the model shown in Figure 2.1. To define the total loss function, we first define the following cross-entropy loss functions, corresponding to the three sub-networks in our model:

$\mathcal{L}_{prediction} = \min_P \mathcal{L}(y_s, P(T(x)))$, where y_s is the true class label of the transform instance x .

$\mathcal{L}_{domain} = \min_D \mathcal{L}(TDL, D(T(x)))$, where TDL is the true domain label, i.e., S for *source*, and T for *target*.

$\mathcal{L}_{transfer} = \min_T \mathcal{L}(FDL, D(T(x)))$, where FDL is a fake domain label, i.e., S for *target*, and T for *source*.

The total loss is defined as follows: $\mathcal{L}_{total} = \mathcal{L}_{domain} + \mathcal{L}_{transfer} + \mathcal{L}_{prediction}$

Let θ_D denote all the parameters in $D(X)$, θ_T denote all the parameters in $T(X)$, and θ_P denote all the parameters in $P(X)$. At each training step (a pass through the source data), we first compute each loss, \mathcal{L}_{domain} , $\mathcal{L}_{transfer}$, and $\mathcal{L}_{prediction}$, and update the parameters accordingly. However, we notice that the discriminant loss function always ‘defeats’ the transfer loss function. To account for this, we compute the transfer loss function two times in each step, as suggested in [72]. Then, the transfer and domain losses become almost equal, which means the transfer function reduced the difference between the two domains to an extent that the discriminant function can not separate the domains. The update rules are shown below:

$$\theta_D \leftarrow \theta_D - \mu \frac{\partial \mathcal{L}_{domain}}{\partial \theta_D} \quad (1)$$

$$\theta_T \leftarrow \theta_T - \mu \frac{\partial \mathcal{L}_{transfer}}{\partial \theta_T} \quad (2)$$

$$\theta_T \leftarrow \theta_T - \mu \frac{\partial \mathcal{L}_{prediction}}{\partial \theta_T} \quad (3)$$

$$\theta_P \leftarrow \theta_P - \mu \frac{\partial \mathcal{L}_{prediction}}{\partial \theta_P} \quad (4)$$

where μ is the learning rate. We use the stochastic gradient descent (SGD) to optimize the parameters. The pseudocode for training the DANN network using SGD is shown in Algorithm 1. Parameter updates are performed for each batch sampled from source. Corresponding to a batch from source of size m (line 5 in the algorithm), which is used to train the prediction network, the algorithm also creates a batch that has half (i.e., $m/2$) source instances and half target instances (line 6 in the algorithm). The source/target batch is used to train the transfer and domain networks.

Algorithm 1 DANN Training Using SGD

Input: Datasets $S = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ (where n_s represents the number of instances in the source S) and

$T = \{x_i^t\}_{i=1}^{n_t}$ (where n_t represents the number of instances in the target T)

Output: $\theta_T, \theta_P, \theta_D$

- 1: $\theta_T \leftarrow$ pre-trained VGG-19 weights
 - 2: $\theta_P, \theta_D \leftarrow$ random initial weights
 - 3: *Step* $k = 0$
 - 4: **for** k from 1 to *steps* **do**
 - 5: **for** each source batch $\{(x_i^s, y_i^s)\}_{i=1}^m$ **do**
 - 6: Sample source/target batch $(x_j^s, x_j^t)_{j=1}^{m/2}$
 - 7: $TDL_j = 1$ if $x_j \in$ source, 0 otherwise
 - 8: $FDL_j = 0$ if $x_j \in$ source, 1 otherwise
 - 9: Compute the loss \mathcal{L}_{domain}
 - 10: Update θ_D using Equation (1)
 - 11: Compute the loss $\mathcal{L}_{transfer}$
 - 12: Update θ_T using Equation (2)
 - 13: Compute the loss $\mathcal{L}_{transfer}$
 - 14: Update θ_T using Equation (2)
 - 15: Compute the loss $\mathcal{L}_{prediction}$
 - 16: Update θ_T and θ_P using Equations (3) and (4)
-

2.4 Experimental Setup

In this section, we present the experimental setup, including the dataset used, cross-validation setup, baselines, and implementation details. The experiments conducted in this study were designed to answer the following research questions: (1) How does the DANN model work for adaptation between disasters of the same type (e.g., two earthquakes) versus adaptation between disasters of different types (e.g., an earthquake and a hurricane)? (2) Does the source-target feature adaptation through DANN give better results as compared to the direct source adaptation from VGG-19? (3) How do the results of the DANN model compare to the results of a VGG-19 network adapted based on data from the target domain itself?

Table 2.1: Disaster class distribution, together with the total number of labeled images in each disaster.

Class	Nepal Earthquake	Ecuador Earthquake	Ruby Typhoon	Matthew Hurricane
Damage	11,183	933	433	204
No-damage	7,919	791	400	127
Total	19,102	1,724	833	331

2.4.1 Dataset

We used the disaster image dataset published by Nguyen et al. [73] in our experiments. The dataset contains images from four disasters, specifically Nepal Earthquake, Ecuador Earthquake, Ruby Typhoon, and Matthew Hurricane. In the original dataset, there are three damage classes: *severe*, *mild*, and *none*. However, Nguyen et al. [73] suggested that the task of discriminating between *mild* and *severe* damage is very subjective, and there is significant overlap in the dataset between the two classes. Therefore, we combine the classes *severe* and *mild* into one class called *damage*. Our goal is to identify images that include damage and separate them from images that do not show any damage. Table 2.1 shows the total number of images in each disaster, and also the class distribution for each disaster.

2.4.2 Setup

Given the four disasters in our dataset, we experiment with all possible source/target pairs. Thus, some source/target pairs consist of disasters of the same type (e.g., two earthquakes), while other pairs consist of disasters of different types (e.g., an earthquake and a hurricane). Furthermore, some pairs have smaller amounts of source data, while others have larger amounts of source data. Similarly, some pairs have a smaller amount of target unlabeled data, while other pairs have a larger amount of target unlabeled data. This variety of pairs makes it possible to answer the research questions we have raised.

For each source/target experiment, we randomly split the target data into target unlabeled (80%) and target test (20%). Each DANN model was trained on all source data and the target unlabeled data, and subsequently tested on the test dataset. The results are eval-

uated using four standard metrics, specifically, accuracy, precision, recall and F1-measure. For cross-validation purposes, we created three different random splits for each source/target pair. The metrics reported represent averages over the three random splits (together with the standard deviation).

2.4.3 Baselines

We compared the DANN model against two baselines: 1) a model that uses only the labeled source data to adapt the pre-trained VGG-19 – this model can generally be seen as a lower bound for domain adaptation; 2) a model that uses the target data as labeled data to adapt the pre-trained VGG-19 – this model can generally be seen as an upper bound for domain adaptation.

2.4.4 Implementation Details

We used TensorFlow’s MomentumOptimizer procedure to train the model using the mini-batch stochastic gradient descent on a K80 graphics card. Based on our preliminary experimentation, we set the learning rate to 0.001, and the batch size to 64 images. Furthermore, we used the dropout technique with a rate of 0.5 to prevent overfitting. The code for the VGG-19 model was adapted from <https://github.com/machrisaa/tensorflow-vgg>.

2.5 Experimental Results

The results of the experiments are shown in Table 2.2 for the DANN model, which was trained on source labeled and target unlabeled data (denoted as DANN-SL-TU), and for the two baselines, VGG-19-SL (VGG-19 adapted using source labeled data), and VGG-19-TL (VGG-19 adapted using target labeled data). The DANN model and the baselines are tested on the same target test data. Each experiment was repeated three times with different target unlabeled/test splits, and the results averaged over the three runs.

Table 2.2: Classification results: average accuracy (Acc.), precision (Prec.), recall (Rec.) and F1-measure (F1) for the damage class (average is taken over three runs), together with the corresponding standard deviation (in parentheses), for DANN-SL-TU (which uses source labeled data and target unlabeled data), VGG-19-SL (adapted using source labeled data), and VGG-19-TL (adapted using target labeled data). Highlighted in bold font are results that are statistically different when comparing VGG-19-SL and DANN-SL-TU using a paired t-test with $p \leq 0.05$.

Source	Ecuador Earthquake											
Target	Matthew Hurricane				Nepal Earthquake				Ruby Typhoon			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
VGG-19-SL	0.557 (0.023)	0.786 (0.058)	0.382 (0.014)	0.514 (0.017)	0.828 (0.006)	0.861 (0.004)	0.841 (0.012)	0.851 (0.006)	0.625 (0.023)	0.823 (0.074)	0.360 (0.048)	0.499 (0.043)
VGG-19-TL	0.821 (0.015)	0.823 (0.026)	0.902 (0.000)	0.859 (0.011)	0.897 (0.005)	0.911 (0.019)	0.916 (0.034)	0.912 (0.005)	0.864 (0.012)	0.872 (0.029)	0.896 (0.049)	0.872 (0.011)
DANN-SL-TU	0.687 (0.026)	0.791 (0.089)	0.683 (0.085)	0.726 (0.023)	0.820 (0.006)	0.842 (0.025)	0.840 (0.013)	0.726 (0.023)	0.741 (0.027)	0.774 (0.072)	0.728 (0.093)	0.744 (0.031)

Source	Nepal Earthquake											
Target	Matthew Hurricane				Ecuador Earthquake				Ruby Typhoon			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
VGG-19-SL	0.642 (0.052)	0.916 (0.035)	0.455 (0.078)	0.606 (0.077)	0.873 (0.006)	0.908 (0.026)	0.852 (0.026)	0.878 (0.006)	0.699 (0.039)	0.890 (0.036)	0.483 (0.078)	0.622 (0.077)
VGG-19-TL	0.821 (0.015)	0.823 (0.026)	0.902 (0.000)	0.859 (0.011)	0.923 (0.010)	0.908 (0.040)	0.957 (0.023)	0.930 (0.008)	0.864 (0.012)	0.872 (0.029)	0.896 (0.049)	0.872 (0.011)
DANN-SL-TU	0.706 (0.023)	0.860 (0.086)	0.634 (0.088)	0.724 (0.038)	0.871 (0.023)	0.860 (0.025)	0.909 (0.038)	0.884 (0.022)	0.800 (0.042)	0.808 (0.043)	0.812 (0.065)	0.819 (0.042)

Source	Matthew Hurricane											
Target	Ecuador Earthquake				Nepal Earthquake				Ruby Typhoon			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
VGG-19-SL	0.747 (0.009)	0.744 (0.013)	0.813 (0.006)	0.777 (0.005)	0.760 (0.009)	0.788 (0.008)	0.808 (0.036)	0.797 (0.014)	0.683 (0.006)	0.650 (0.011)	0.851 (0.023)	0.736 (0.002)
VGG-19-TL	0.923 (0.010)	0.908 (0.040)	0.957 (0.023)	0.930 (0.008)	0.897 (0.005)	0.911 (0.019)	0.916 (0.034)	0.912 (0.005)	0.864 (0.012)	0.872 (0.029)	0.896 (0.049)	0.872 (0.011)
DANN-SL-TU	0.761 (0.038)	0.747 (0.054)	0.870 (0.037)	0.802 (0.023)	0.755 (0.023)	0.784 (0.012)	0.805 (0.073)	0.793 (0.030)	0.681 (0.024)	0.669 (0.038)	0.774 (0.103)	0.713 (0.032)

Source	Ruby Typhoon											
Target	Matthew Hurricane				Nepal Earthquake				Ecuador Earthquake			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
VGG-19-SL	0.741 (0.018)	0.817 (0.016)	0.748 (0.027)	0.778 (0.022)	0.731 (0.009)	0.849 (0.010)	0.658 (0.008)	0.741 (0.008)	0.740 (0.018)	0.865 (0.016)	0.618 (0.027)	0.721 (0.022)
VGG-19-TL	0.821 (0.015)	0.823 (0.026)	0.902 (0.000)	0.859 (0.011)	0.897 (0.005)	0.911 (0.019)	0.916 (0.034)	0.912 (0.005)	0.923 (0.010)	0.908 (0.040)	0.957 (0.023)	0.930 (0.008)
DANN-SL-TU	0.736 (0.017)	0.818 (0.057)	0.740 (0.051)	0.774 (0.009)	0.763 (0.021)	0.813 (0.023)	0.777 (0.083)	0.792 (0.033)	0.809 (0.032)	0.814 (0.024)	0.840 (0.080)	0.825 (0.037)

2.5.1 Discussion of the Results

To answer our first research question, *How does the DANN model work for adaptation between disasters of the same type versus adaptation between disasters of different types?*, we compare the results of DANN-SL-TU with the results of VGG-19-SL for pairs of similar and different disasters. We consider Matthew Hurricane and Ruby Typhoon to be disasters of the same type, and the same for Ecuador Earthquake and Nepal Earthquake. Disaster pairs consisting of a hurricane/typhoon and an earthquake are considered to be different. As can be seen from Table 2.2, the results of DANN-SL-TU are similar and sometimes better than the results of VGG-19-SL, for pairs of similar disasters. Furthermore, the DANN-SL-TU results are generally better than the VGG-19-SL results for different disasters. For example, when we use either Ecuador Earthquake or Nepal Earthquake as source, the results for predicting the other earthquake are similar between DANN-SL-TU and VGG-19-SL. However, if we use either disaster as source, and predict Matthew Hurricane or Ruby Typhoon, DANN-SL-TU gives significantly better results. Similarly, when Ruby Typhoon is used as source, the results obtained with DANN-SL-TU for Nepal Earthquake and Ecuador Earthquake are significantly better than the results obtained with VGG-19-SL, while the two models are similar for Matthew Hurricane.

Given the above-mentioned results, the answer to our second question, *Does the source-target feature adaptation through DANN give better results as compared to the direct source adaptation from VGG-19?*, is overall positive, as the DANN-SL-TU model gives similar or better results than VGG-19-SL. However, if we consider the size of the datasets used in our experiments, the results in Table 2.2 suggest that the DANN-SL-TU approach gives better results if the source labeled data and the target unlabeled data used for training are relatively large (e.g., a few thousands). Furthermore, it can be seen that DANN increased the recall for the damage class in most case, especially when the source and target disasters are of different types. Finally, when comparing DANN-SL-TU with VGG-19-TL to answer our third question, *How do the results of the DANN model compare to the results of a VGG-19 network adapted based on data from the target domain itself?*, we observe that indeed VGG-

19-TL acts as an upper bound for the DANN-SL-TU model. While the gap between the two models is significant, the gap between VGG-19-TL and VGG-19-SL is also large, suggesting that the source data distribution is indeed different from the target data distribution, and better domain adaptation models have the potential to bridge the gap.

Figure 2.2 shows examples of images that are correctly classified or miss-classified by the DANN-SL-TU/VGG-19-SL/VGG-19-TL networks, when Ruby Typhoon is used as target disaster and Ecuador Earthquake is used as source disasters. Specifically, the top row in Figure 2.2 shows examples of images that are correctly classified as *damage* by DANN-SL-TU and VGG-19-TL, but miss-classified as *no-damage* by VGG-19-SL. As the VGG-19-SL network is trained to recognize earthquake damage, it can fail to recognize rain/flood/wind damage present in typhoon images. As opposed to that, the middle row in Figure 2.2 shows images that are miss-classified as *damage* by both DANN-SL-TU and VGG-19-TL, but correctly classified as *no-damage* by VGG-19-SL. It seems that the networks that use images from the target typhoon during training may over-learn that water-related or wind-like images are showing damage. Finally, the bottom row in Figure 2.2 shows images that are miss-classified by both DANN-SL-TU and VGG-19-SL. The first two images in the bottom row are correctly classified by VGG-19-TL as *damage* and *no-damage*, respectively, while the last two images are also miss-classified by VGG-19-TL as *damage*. Together with the quantitative evaluation in Table 2.2, these examples suggest that more unlabeled images from the target, and ideally more labeled images from the source, can potentially improve the results.

2.5.2 Estimating the Distance between Source and Target Domains before and after Adaptation

To evaluate the difficulty of the domain adaptation tasks in our study, we used the proxy \mathcal{A} -distance [74] to measure the domain shift. We expect domain adaptation to give better results than VGG-19-SL if the source and target domains are different, and similar results if the target and source domains are similar. Table 2.3 shows the domain pairs used in our



Figure 2.2: Examples of images that are correctly classified or miss-classified by different networks, when Ruby Typhoon is used as target disaster and Ecuador Earthquake is used as source disaster. (Top) Images that are classified correctly as *damage* by DANN-SL-TU and VGG-19-TL, and miss-classified as *no-damage* by VGG-19-SL. (Middle) Images that are miss-classified as *damage* by DANN-SL-TU and VGG-19-TL, while VGG-19-SL correctly classifies them as *no-damage*. (Bottom) The first two images on the left are miss-classified by both DANN-SL-TU and VGG-19-SL, while VGG-19-TL correctly classifies image (h) as *damage* and image (i) as *no-damage*. The last two images on the right have label *no-damage* and are miss-classified by all networks as *damage*.

Table 2.3: Proxy \mathcal{A} -distance between different domains. In the ‘After Adaptation’ case, $(A \rightarrow B)$ means that the model was trained with A as source and B as target, while $(B \rightarrow A)$ means that the model was trained with B as source and A as target.

Domain A	Domain B	Before Adaptation	After Adaptation ($A \rightarrow B$)	After Adaptation ($B \rightarrow A$)
Matthew Hurricane	Ruby Typhoon	1.287	0.625	0.141
Ecuador Earthquake	Matthew Hurricane	1.633	0.125	0.141
Ecuador Earthquake	Ruby Typhoon	1.637	0.250	0.219
Nepal Earthquake	Ecuador Earthquake	1.831	0.132	0.091
Nepal Earthquake	Matthew Hurricane	1.951	0.078	0.281
Nepal Earthquake	Ruby Typhoon	1.915	0.112	0.288

experiments and their corresponding domain divergence in terms of the proxy \mathcal{A} -distance, before and after feature adaptation. To compute the proxy \mathcal{A} -distance, we trained a CNN model to separate the source domain X_S from the target domain X_T using unlabeled data. Then, we calculated the proxy \mathcal{A} -distance as $2(1 - 2\epsilon)$, where ϵ is the mis-classification error on test data. A high proxy \mathcal{A} -distance means that the two domains are far apart, and benefit from adaptation. A proxy \mathcal{A} -distance close to zero means that the two domains have essentially the same distribution. In addition to the proxy \mathcal{A} -distance before adaptation, we also calculated the proxy \mathcal{A} -distance after adaptation to understand if domains with different distributions are brought closer through adaptation. We should note that the model used to obtain the proxy \mathcal{A} -distance between $T(X_S)$ and $T(X_T)$ is equivalent with a two layer perceptron network, based on our model architecture. In practice, we found that sometimes the classification error may be larger than 0.5, which resulted in a negative value for the proxy \mathcal{A} -distance. In such cases, we took the absolute value of the proxy \mathcal{A} -distance, as suggested in [74]. From Table 2.3, we can see that the distance between Matthew Hurricane and Typhoon Ruby is the smallest among all the distances, suggesting that the distributions of the two disasters are similar. Also, considering the pairs which contain Nepal Earthquake, we can see that the smallest distance is obtained for Ecuador Earthquake. Thus, the distances in Table 2.3 generally are in agreement with the classification results in Table 2.2. It is also interesting to note that, after domain adaptation, all proxy \mathcal{A} -distances become smaller, which means that the distributions of $T(X_S)$ and $T(X_T)$ are more similar as compared to the original distributions of the source and target datasets.

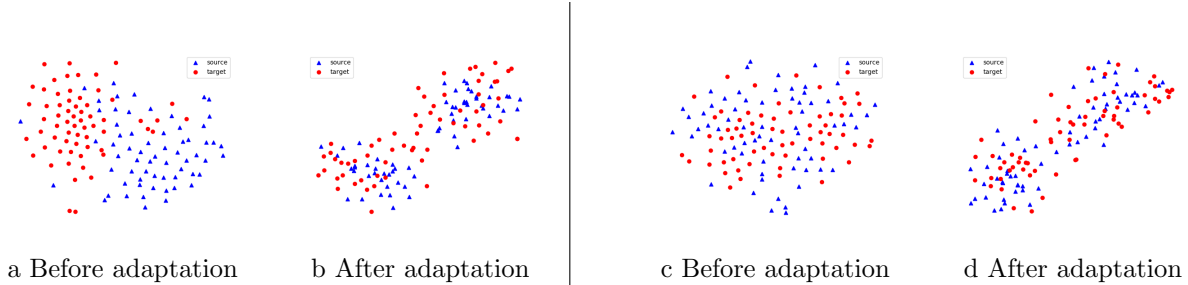


Figure 2.3: Two dimensional representation of source (blue) and target (red) using the VGG-19-SL representation of the images (before adaptation), and the DANN-SL-TU transformed representation of the images (after adaptation), for two pairs of disasters. The pair on the left consists of two dissimilar disasters, Ecuador Earthquake (source, blue) and Ruby Typhoon (target, red). The pair on the right consists of two similar disasters, Ecuador Earthquake (source, blue) and Nepal Earthquake (target, red). The two representations are reduced to 2 dimensions using the t-SNE [1] technique.

2.5.3 Visual Distribution Analysis

It can be proven that $T(X_S)$ and $T(X_T)$ have similar distributions after the domain adversarial training. Here, we visualize $T(X_S)$ and $T(X_T)$ for some sample source/target pairs to visually analyze the distributions of the transformed source and target data, by comparison with the distribution of the original source and target data.

To perform this visualization, first, we randomly selected 64 images from Ruby Typhoon and 64 images from Ecuador Earthquake, two disasters with relatively high distance. Using the VGG-19 model trained on Ecuador Earthquake, we extracted the weights of the last fully connected layer in the VGG-19 model (this layer has dimension 4096). Next, we used t-SNE [1] to reduce the dimensionality from 4096 to 2. As a result, each image is represented by a two dimensional vector. We plotted the 128 images using the binary representation in Figure 2.3 (a), where we used blue triangles for images from Ecuador Earthquake and red dots for images from Ruby Typhoon. Similarly, using the DANN model trained on Ecuador Earthquake as source and Ruby Typhoon as target, we extracted the weights of the first fully connected layer, which represents the output of the transfer function $T(x)$. The graph for the 128 images (from Ecuador Earthquake and Ruby Typhoon) is shown in Figure 2.3 (b), where the blue triangles represent images from Ecuador Earthquake and the red dots represent images from Ruby Typhoon. As can be seen, the graph in Figure

2.3 (a) shows a clear boundary between the two domains, while in the graph in Figure 2.3 (b) the two domains are harder to separate as their distributions overlap significantly. Thus, the visualization of the reduced representations shows that after domain adaptation, the difference between source (Ecuador Earthquake) and target (Ruby Typhoon) has been indeed reduced.

In addition to performing the visualization of the transformed representations for two domains with high distance, we also performed it for two domains with relatively smaller distance. Specifically, we randomly selected 64 images from Nepal Earthquake and 64 images from Ecuador Earthquake. Following the same procedure described above, we plot two graphs in Figure 2.3 (c) and (d), respectively. In (c), the VGG-19 model was trained on Ecuador Earthquake and in (d), the DANN model was trained on Ecuador Earthquake as source and Nepal Earthquake as target. As can be seen in (c), the distributions of the two domains are indistinguishable even before adaptation, as the two domains have similar distributions in the first place. Thus, in this situation, DANN can not improve much the results of VGG-19, as the domains are already similar.

2.6 Conclusions and Discussion

In this paper, we studied the application of domain adaptation to identify disaster images that show damage. We adapted the DANN approach by combining it with the VGG-19 model, and thus taking advantage of the available labeled data used to train VGG-19. Experimental results suggest that the domain adaptation approach is especially useful when the source and target disasters are of different types, and thousands of labeled instances are available for the source disaster (a requirement which can be met without much effort or cost).

Given that the proposed approach does not require labeled data from the target domain, it can potentially be used to identify damage images in nearly real time. To do this, one would first use a pre-trained model that identifies images relevant to disasters and filters out non-relevant images. A model like this is already available as part of the AIDR platform

(<http://aidr.qcri.org>), as described by Nguyen et al. [73]. Subsequently, our approach can be used to identify images that show damage, by first employing the model pre-trained on source data, and gradually adapting it based on unlabeled disaster-relevant target images. The identified damage images could be used to perform fast damage assessment, which in turn can result in faster disaster response. Thus, we envision that our approach, together with relevancy-filtering models, can be incorporated in the routine operations of disaster response and management organization.

We should note that our current models were trained to distinguish between damage and no-damage images. This is a limitation imposed by the data that was available for training, which did not discriminate well between severe and mild damage classes. However, as labeled images for more specific classes of damage become available, our models can be re-trained and adapted to identify such classes. Moreover, one can further analyze images that show damage by localizing and quantifying damage on a continuous scale, as shown in Li et al. [40]. Such level of detail can help response teams prioritize their operations based on the severity of the damage, assuming that resources are scarce.

As part of future work, we plan to apply the DANN approach to other disaster image classification problems, for example, specific types of damage including building damage, road damage, bridge damage, human damage, etc. We also plan to study other domain adaptation approaches in the context of image analysis for disaster response. Finally, we aim to study approaches that can help identify image features indicative of damage, or even other features that might be useful for response teams.

Chapter 3

Domain Adaptation with Reconstruction on Disaster Tweets

3.1 Introduction

In recent years, social media platforms, such as Twitter or Facebook, have become effective communication tools, complementing more traditional communication networks (e.g., 9-1-1 centers), which are not always well-equipped to handle the large volume of requests posted by disaster-affected individuals [75, 47]. Early works in disaster management have discussed the usefulness of social media data in improving disaster resilience [5, 76, 77], situational awareness [78], and emergency response [79]. To facilitate the use of social media data posted by eyewitnesses of disasters, machine learning approaches have been developed to filter informative tweets and classify them with respect to various situational awareness categories [80, 81, 82]. Many of the existing studies have employed supervised learning to train a classifier on data from a disaster of interest, and used the classifier to make predictions on data from the same disaster. In supervised learning, data need to be manually labeled before a classifier can be trained. The data labeling process is usually expensive and time-consuming, making it hard to use supervised learning in the beginning of a disaster.

However, unlabeled data from a target disaster accumulates quickly. In addition, labeled

data from a prior source disaster is readily available. A supervised classifier learned on the source disaster may not perform well on the target disaster due to differences in the distributions of the two disasters (a.k.a., covariate shift). Domain adaptation approaches that use the labeled source disaster data, together with unlabeled target disaster data, to train a classifier on source and adapt it on target, represent a promising solution to the filtering of information posted on social media during disasters. By eliminating the need for labeled target data, domain adaptation approaches can potentially improve the response time especially in the beginning of a disaster. While domain adaptation has produced promising results on this task [10, 11, 12, 2],[37], the potential of domain adaptation approaches based on deep learning is largely unexplored.

One challenge for domain adaptation is how to effectively use the information from unlabeled target data to adapt a classifier trained on labeled source data [83]. There are several popular approaches that can be used to address this challenge. One idea is to reduce the shift between the source and target data distributions. Domain adversarial neural networks (DANN) [32] implement this idea by co-training a feature extraction network together with a source classification network. The feature extraction network aims to transform the source and target features to make them indistinguishable. Given domain invariant features, the source classification network trained on labeled source data can be effectively used to classify target data. Another way to reduce the covariate shift is to utilize domain reconstruction-classification networks (DRCN). The reconstruction network consists of an autoencoder (i.e., encoder-decoder) that reconstructs the target data in a self-supervised manner [36]. The autoencoder network is co-trained with a source classification network that shares the encoder component of the autoencoder. Being shared between the two networks, the encoder will learn information from both source and target data. As a consequence, the source classification network, trained on labeled source data, can be used to classify target data.

In the context of disaster tweet classification, Alam et al. [2] proposed a domain adaptation approach which combines domain adversarial training and graph embeddings with a classification network. The adversarial training is used to reduce the distribution shift, while the graph embeddings are used to induce structural similarity between source and target

data. [37] proposed an adaptation approach based on a domain adversarial neural network, with a large convolutional neural network (CNN) as its backbone, to identify disaster images that show damage. To the best of our knowledge, there is no study on the use of domain reconstruction-classification networks on social media disaster data.

To fill in this gap, in Section 3.2, we propose a domain adaptation approach for tweet classification, which implements the domain reconstruction-classification idea using a sequence-to-sequence autoencoder [84]. Both the encoder and the decoder components of the autoencoder represent recurrent neural networks (RNN), specifically long short-term memory (LSTM) networks. Our experimental results in Section 3.3 show that the proposed domain reconstruction-classification approach outperforms the existing approach based on domain adversarial networks and graph embeddings [2]. We conclude the paper and present ideas for future work in Section 3.5.

3.2 Approach

In this section, we describe the components of the proposed approach. The overall architecture of the approach is shown in Fig. 3.1.

3.2.1 Domain Adaptation

Let $X_s = \{(X_i^s, y_i^s)_{i=1}^{n_s}\}$ be a labeled source dataset, where n_s represents the number of instances in the source dataset, X_i^s contains a sequence of words $X_i^s = \{x_j\}_{j=1}^{|X_i^s|}$ and y_i^s is the sequence label. Let $X_t = \{(X_i^t)_{i=1}^{n_t}\}$ be an unlabeled target dataset, where n_t represents the number of instances in the target dataset, and X_t contains a sequence of words $X_i^t = \{x_j\}_{j=1}^{|X_i^t|}$. The task in domain adaptation is to learn a classifier for the target domain using the labeled source data, X^s , together with unlabeled target data X^t . The source data and target data may share some patterns but generally have different distributions. For example, the source data may consist of tweets posted during an earthquake, while the target data may consist of tweets posted during a hurricane.

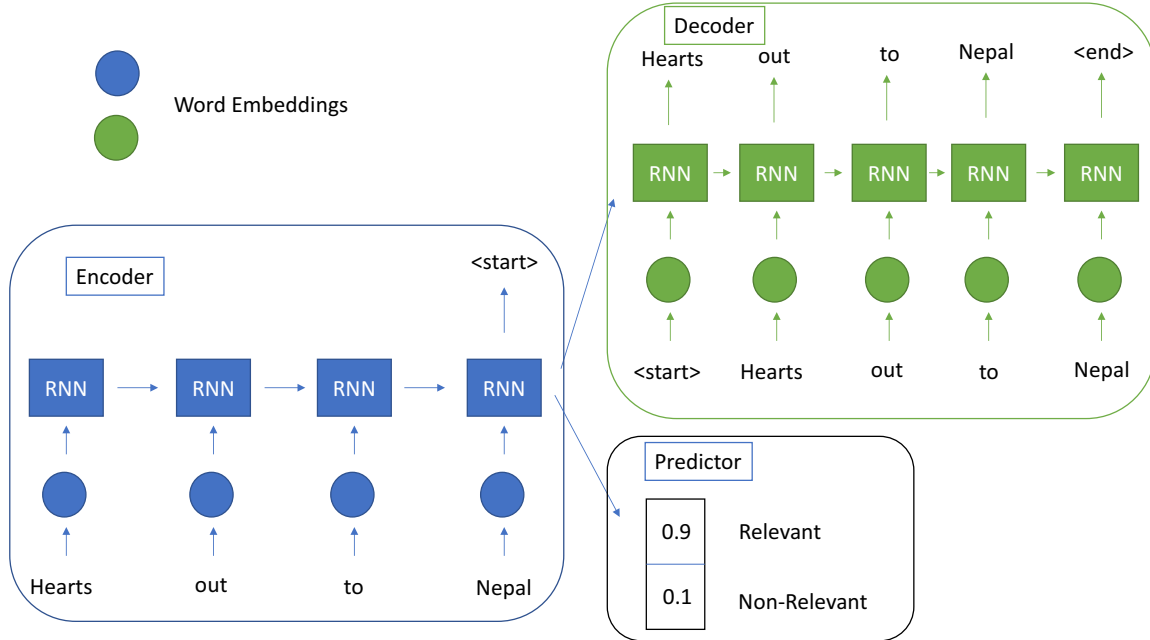


Figure 3.1: Architecture of the proposed approach

3.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) have been widely used for many tasks in Natural Language Processing, as they can capture dependencies in sequence data. Among other tasks, RNNs can be used to classify text. Formally, given a sequence (x_1, x_2, \dots, x_n) , an RNN model can predict the label y using the following forward equations:

$$\begin{aligned}
 h_i &= RNN(x_i, h_{i-1}) \quad i \in 1, \dots, n \\
 y &= RNN(h_n)
 \end{aligned}
 \tag{3.1}$$

where h_i represents the hidden state at time step i and acts as the memory of the network (h_0 is initialized with a zero vector).

3.2.3 Seq2Seq Autoencoder

The seq2seq model was proposed by Sutskever et al. [84] in the context of machine translation. It uses an RNN network to encode an original sequence into a hidden vector and

then uses another RNN network to decode the hidden vector into another output sequence. If the original sequence and the output sequence are from different languages, the task achieved is translation. If the original sequence and the output sequence are the same, the task corresponds to autoencoding, and the model is called an autoencoder. An autoencoder can produce a sequence representation in a self-supervised manner (a.k.a., unsupervised learning). The representation retains all the information needed to reconstruct the original sequence itself.

3.2.4 Domain Adaptation with Reconstruction

Our proposed domain adaptation with reconstruction model combines the techniques discussed above. For the source data X_s , we initialize a set of word embedding w_s corresponding to words x_s in a sequence X_i^s , and then train network to predict the label for X_i^s :

$$\hat{y}_i^s = FC(Encoder(X_i^s)) \quad (3.2)$$

where the *Encoder* is an RNN network which extracts the encoded representation of the sequence X_i^s , and *FC* is a fully-connected layer which predicts the label of X_i^s using the representation. For the target data X_i^t , we train a **seq2seq** autoencoder on the initialized word embeddings. Formally, for each target instance X_i^t , we have:

$$\hat{X}_i^t = Decoder(Encoder(X_i^t))$$

The **seq2seq** model and the RNN classification network share the encoder layer. Thus, the RNN encoder layer learns a good classifier on the labeled source data while using information from the unlabeled target data. The model will be trained by optimize

$$L = \min(\mathcal{L}(y_i^s, \hat{y}_s) + \mathcal{L}(X_i^t, \hat{X}_i^t))$$

where \mathcal{L} is cross-entropy loss. The pseudocode for the training phase of the proposed model is shown in Algorithm 2.

Algorithm 2 Model Training Using SGD

Input: Source $X_s = \{(X_i^s, y_i^s)_{i=1}^{n_s}\}$, and target $X_t = \{(X_i^t)_{i=1}^{n_t}\}$

Output: $\theta_E, \theta_D, \theta_P$ (where $\theta_E, \theta_D, \theta_P$ represent the parameters in the Encoder, Decoder and Predictor components, shown in Fig. 3.1).

- 1: $\theta_E, \theta_D, \theta_P \leftarrow$ random initial weights
 - 2: **for** k from 1 to *steps* **do**
 - 3: **for** each source batch $\{(X_i^s, y_i^s)\}_{i=1}^m$ **do**
 - 4: Sample target unlabeled batch $(X_j^t)_{j=1}^m$
 - 5: Update θ_E, θ_P to minimize the loss $\mathcal{L}(y_i^s, \hat{y}_i^s)$
 - 6: Update θ_E, θ_P to minimize the loss $\mathcal{L}(X_i^t, \hat{X}_i^t)$
-

3.3 Experimental Setup

3.3.1 Implementation Details

The implementation details of our model are as follows: In the `seq2seq` model implementation, both the encoder and the decoder contain one layer of RNN cells. The dimension of the hidden vector representation is $p = 200$. We use the Adam optimizer to train the model with a learning rate of 0.001 and a mini-batch of size 64.

3.3.2 Datasets

The first dataset used in this study was published by Alam et al. [2], who made available the *train*, *validation* and *test* splits used in their study. The dataset contains tweets crawled during Queensland Floods (QFL) and Nepal Earthquake (NEQ). Each tweet was labeled as *relevant* or *non-relevant*. Statistics about the dataset and the *train*, *validation* and *test* splits are shown in Table 3.1. In our experiments, for a source-target pair, we used the *train* split of the source data to train the RNN classification network, we used the *validation* split of

the target data to train the `seq2seq` autoencoder, and finally the *test* split of the target data to evaluate the model.

Dataset	Relevant	Non-Relevant	Train	Validation	Test
NEQ	5527	6141	7000	1167	3503
QFL	5414	4619	6019	1003	3011

Table 3.1: Statistics of the dataset published in [2]

The second dataset used (called CrisisMMD) was published by Alam et al. [3]. It contains tweets crawled during 7 disaster events. Each tweet was labeled as *informative* or *non-informative*. Statistics about the CrisisMMD dataset are shown in Table 3.2. For each source-target experiment, we randomly split the target data into 70% *unlabeled* and 30% *test*. We used all the labeled source data to train the RNN classification network. The target *unlabeled* data is used to train the autoencoder, while the target *test* data is used to evaluate the model. We conduct experiments where the datasets D0, D1, D2, D3, D5 are used as source datasets, respectively. We didn't use the datasets D4 and D6 as sources, as they are relatively small and unbalanced compared to the other datasets. In addition to using one disaster as source, we also performed experiments where we used *all-but-one* disasters as source and the remaining disaster as target.

	Dataset	Not-Informative	Informative	Total
D0	California Fire	324	1162	1486
D1	Hurricane Harvey	973	3027	4000
D2	Hurricane Irma	836	3201	4037
D3	Hurricane Maria	1490	2510	4000
D4	Iraq Iran Earthquake	85	402	487
D5	Mexico Earthquake	314	925	1239
D6	Srilanka Floods	544	287	831

Table 3.2: Statistics of the the CrisisMMD dataset [3]

3.3.3 Research Questions

Our experiments are designed to answer the following questions:

- R1 How do the results of our domain adaptation with reconstruction approach compare with the results of the domain adversarial with graph embeddings approach [2]?
- R2 How do the results of our domain adaptation with reconstruction approach compare with the results of the corresponding *cross-domain* supervised RNN classification models?
- R3 How do the results of the experiments that use one disaster as source compare with the results of the experiments that use the union of *all-but-one* disaster as source?

3.4 Experimental Results

3.4.1 Results

The results of the experiments on the dataset introduced in [2] are presented in Table 3.3. In addition to the results of the RNN and RNN+AE models (in domain and cross-domain), Table 3.3 also shows the results of the models in [2]. As can be seen, the RNN classification model is better than the supervised model used in [2] in both cross-domain (model trained on source) and in-domain (model trained on target) experiments. The RNN model is also better than the adaptation model proposed by Alam et al. [2] by a large margin. Furthermore, the table shows that the domain adaptation with reconstruction model achieves better results than the RNN model alone, even when used for in-domain experiments (where both the RNN network and the autoencoder are trained on the target data).

The results of the experiments on the CrisisMMD dataset [3] are presented in Table 3.4. We show the average accuracy, AUC and F1 score over five independent runs. Each row has same target data, while each column has same source data (except for the last column that shows the *all-but-one* results). The table also shows in-domain results (where the models are trained on target). As can be seen from the table, the results of the domain adaptation with reconstruction model are better than the results of the RNN network in most cases (including for in-domain models). While the adaptation from one source disaster to the

Model (cross-domain)	Acc	AUC	F₁	Acc	AUC	F₁
source -> target	QFL->NEQ			NEQ->QFL		
supervised [2]	n/a	54.86	53.63	n/a	58.99	59.10
domain adv. (DA) [2]	n/a	57.63	57.79	n/a	60.15	60.94
graph emb. (GE) [2]	n/a	54.60	54.79	n/a	60.38	60.54
GE + DA [2]	n/a	58.81	59.05	n/a	66.49	65.92
RNN	64.19	65.42	64.18	55.83	61.36	55.17
RNN+AE (our)	68.82	72.29	68.38	81.24	87.79	81.18
Model (in-domain)	Acc	AUC	F₁	Acc	AUC	F₁
target	NEQ			QFL		
supervised [2]	n/a	61.22	60.89	n/a	80.14	80.16
RNN	69.99	76.35	69.97	93.09	97.71	93.07
RNN+AE	72.39	79.55	72.40	93.89	97.18	93.89
Bert	78.54	78.71	78.37	96.51	96.39	96.81

Table 3.3: Cross-domain and in-domain results.

target disaster improves the results of the RNN classification network (especially, when the source and target disasters are similar, e.g., both are hurricanes), overall, the *all-but-one* models give the best results for a target.

3.4.2 Discussion

We will use the results of the experiments reported in Tables 3.3 and 3.4 to answer our research questions.

- R1 Our proposed domain adaptation with reconstruction approach has dramatically better performance as compared to the domain adversarial with embeddings, in all metrics considered.
- R2 The domain adaptation with reconstruction has better performance than the cross-domain supervised RNN classification network in most cases.
- R3 Using all the available sources for training (*all-but-one* experiments) with the domain

Method	Acc	AUC	F1	Acc	AUC	F1	Acc	AUC	F1	Acc	AUC	F1	Acc	AUC	F1	Acc	AUC	F1
Pair	D0→D0			D1→D0			D2→D0			D3→D0			D5→D0			all-but-D0→D0		
RNN	75.78	70.36	72.58	69.73	61.69	50.02	73.92	64.41	57.15	71.38	68.67	62.07	64.80	60.15	54.46	72.65	68.54	72.94
RNN+AE	75.11	71.37	72.11	78.25	66.77	52.88	79.45	71.71	60.27	73.99	68.43	60.37	77.88	68.74	57.44	80.27	73.27	76.24
Pair	D0→D1			D1→D1			D2→D1			D3→D1			D5→D1			all-but-D1→D1		
RNN	68.79	62.04	53.46	81.17	82.41	78.92	77.00	76.37	62.27	72.56	75.38	66.47	58.97	62.49	54.52	77.17	78.53	76.33
RNN+AE	72.00	67.79	60.97	80.92	81.84	78.66	77.53	78.56	63.92	78.25	78.37	69.29	70.22	69.46	61.32	76.92	77.79	77.55
Pair	D0→D2			D1→D2			D2→D2			D3→D2			D5→D2			all-but-D2→D2		
RNN	70.27	65.27	57.94	76.28	71.77	61.64	81.29	76.53	78.60	71.75	72.80	63.18	53.15	58.66	49.15	77.74	71.47	76.71
RNN+AE	73.21	69.59	59.87	77.05	75.94	64.14	80.54	76.91	78.13	74.00	75.13	63.64	58.56	64.65	53.65	78.48	75.26	78.68
Pair	D0→D3			D1→D3			D2→D3			D3→D3			D5→D3			all-but-D3→D3		
RNN	60.69	56.91	52.05	66.44	70.59	55.21	67.67	71.51	59.25	72.25	77.47	71.06	54.78	58.19	53.98	68.58	73.31	66.04
RNN+AE	65.36	67.70	59.07	68.53	72.84	60.15	67.14	69.98	56.21	72.00	79.68	72.46	61.33	60.20	52.50	73.17	77.09	72.05
Pair	D0→D4			D1→D4			D2→D4			D3→D4			D5→D4			all-but-D4→D4		
RNN	57.11	59.98	49.85	68.22	65.33	54.43	72.66	53.41	49.41	59.56	65.57	52.45	78.00	67.72	62.32	74.67	78.43	77.07
RNN+AE	77.56	65.77	55.25	81.56	67.29	52.10	79.11	51.39	51.36	81.11	66.97	58.86	77.78	69.21	55.98	79.33	77.31	78.44
Pair	D0→D5			D1→D5			D2→D5			D3→D5			D5→D5			all-but-D5→D5		
RNN	70.34	63.03	59.40	71.69	66.58	57.56	76.34	63.97	56.08	52.78	56.56	50.51	75.27	72.36	74.10	74.19	71.74	74.28
RNN+AE	75.54	70.13	59.65	74.46	73.24	50.82	77.42	68.83	58.38	73.48	70.56	60.54	77.15	76.83	74.91	77.42	73.04	74.45
Pair	D0→D6			D1→D6			D2→D6			D3→D6			D5→D6			all-but-D6→D6		
RNN	57.60	74.67	57.11	69.60	87.45	69.43	63.07	79.78	62.98	73.07	81.31	72.04	53.47	65.61	53.31	74.19	71.74	74.28
RNN+AE	65.73	80.62	65.15	71.33	92.19	71.18	57.60	82.52	57.38	69.07	86.93	68.71	56.00	77.45	55.71	78.00	90.30	78.14

Table 3.4: Cross-domain and in-domain (e.g., D0→D0) Accuracy, AUC and F1 for Crisis-MMD. Results based on 5 independent runs.

adaptation with reconstruction approach gives the best performance overall.

3.5 Conclusions

In this paper, we proposed a domain adaptation approach for disaster tweet classification. The approach works by reconstructing the target data to find an informative hidden representation, and predicting the target label based on a classifier learned from the labeled source data. Experimental results show that the performance of proposed model is better than the performance of another adaptation model recently proposed for this task [2]. Given the promising results, our approach can potentially help eliminate the laborious process of labeling data in the beginning of a disaster. As a consequence, disaster response can be accelerated and resources can be better allocated.

As part of future work, we plan to evaluate other implementations for the Autoencoder, including Transformers, which have been shown to give good results in machine translation. We also plan to experiment with different types of word embedding (e.g., BERT) fine-tuned on disaster data, as an alternative to embeddings which are randomly initialized before training the models.

Chapter 4

Improving Disaster related Tweet Classification with Using a Multimodal Deep Learning Approach

4.1 Introduction



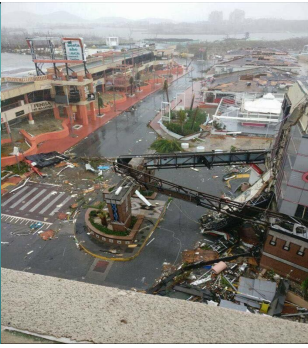
The importance of social media analysis in disaster management has been widely recognized [7, 75, 85, 79]. During a disaster, there is a large volume of communication requests, as people contact families and friends to inform them about disaster developments. Standard communication means may be lost in the beginning of a disaster due to a very heavy workload. This can make the disaster response task very challenging, and presents a need for more efficient communication and data collection tools. Social media has received lots of attention in disaster response. It allows people to share information and to ask for help. Furthermore, it is relatively easy to collect data from social media platforms (e.g., Twitter). If data from social media can be analyzed effectively, the rescue and recovery process can be accelerated because people can be connected with the resources that they need in a timely manner.

There are several studies focused on data collection during a disaster [86, 3]. Such studies

describe detailed steps for data collection and labeling, and make available labeled datasets which can be used for image and/or text classification tasks.

Most of the prior works using such datasets have focused either on text classification [17, 87, 80, 88, 13, 82] or image classification [18, 37]. However, the text and the image of a tweet presumably hold complementary pieces of information, which can be used together to improve the overall classification of disaster tweets. Because of this relationship between

Table 4.1: Tweet text and image examples

Text Label	Informative	Not Informative	Informative
Text	puerto rico could be do n't care if it 's hurricane irma de- without power for hurricane irma , hy- stroys 'number¿ ' ;number¿ to ;num- pothermia , or even of french part of ber¿ months after rae sremmurd . i caribbean island st hurricane irma ;url¿ 'm just here to throw martin : official ;url¿ ;url¿		
Image Label	Not Informative	Not Informative	Informative
Image			

tweet text and images, it is of interest to utilize both text and images together in a model to potentially improve the performance of the models learned from text and/or images separately. Some existing studies proposed to use multimodal data for disaster tweet classification [89, 29], and trained models on tweets for which the text and image labels agree. While it is expected that the two modalities will enhance each other if their labels agree, it is also of interest to study if they can help each other when the labels don't agree.

Thus, in this paper, we propose a multimodal approach for tweet classification, which

makes use of both text and image information in a tweet, when both are available. We evaluate the multimodal approach under two different scenarios. In the first scenario, we only use tweets with matching text and image labels to understand if combining the two modalities results in better classification performance. In the second scenario, we use not only tweets with matching text and image labels, but also tweets with different text and image labels. Here, the goal is to see if one modality helps the classification with respect to the other modality despite potential label mismatches between the two modalities. In addition to understanding the benefits achieved when combining two modalities, we also aim to understand if one modality is more predictive than the other.

The rest of this paper is organized as follows: We describe the proposed model in the “APPROACH” section. We discuss the related work in “RELATED WORK”. Experimental results are presented and discussed in the “EXPERIMENTAL RESULTS” section. Finally, we conclude the paper in the “CONCLUSIONS” section.

4.2 Related Work

Lots of studies focused on supervised learning on tweet texts and images. Chowdhury et al. [90] and Stowe et al. [91] proposed classification models for disaster related tweet classification with traditional machine learning algorithms. Neppalli et al. [88] and Li et al. [13] proposed the use of deep learning models to conduct text classification. Neppalli et al. [88] also compared the performance between traditional Naive Bayes models and deep learning models, while Li et al. [13] compared different word embeddings. There are also some studies focused on disaster image analysis. Yang et al. [92] proposed a hierarchical image classification approach to enhance situational awareness. Barnes et al. [93] and Vetrivel et al. [94] analyzed satellite images to identify blocked routes and potential rescue targets. They also analyzed satellite disaster images to assess disaster damage. An image classification model has been proposed by Nguyen et al. [18], which applied the deep Convolutional Neural Network model on disaster images.

Hu and Flaxman [28] proposed an multimodal approach to predict emotion word tags for posts made by Tumblr users. Our model is similar with their model but applied on disaster-related tweets. A multimodal approach has been published by Mouzannar et al. [29]. They trained a CNN model for text and another CNN model for images, and then took the average of the output to generate the final classification decision. At a high level, our proposed approach is similar to the approach in Mouzannar et al. [29]. However, we use different base models for text and images, respectively, and an improved way of combining the predictions, in addition to the average of the predictions. Nalluru et al. [30] proposed a multimodal model which extracts embeddings for text and images and then applies the lightGBM model, which is an implementation of fast gradient boosting on decision tree [95]. Compared to this work, we use different embedding extraction methods and apply a fully-connected layer for classification (instead of the lightGBM model), as this approach is widely used in deep learning. Agarwal et al. [89] proposed a novel end-to-end multimodal framework. Their framework addresses three classification tasks: tweet informativeness, infrastructure damage and damage severity. They combine cues extracted from text and image modalities, and merge them using the attention mechanism. All of the abovementioned works are used for tweets with matching text and image labels, but do not consider the the scenario where the labels of the text and images may disagree. Our proposed approach can handle both scenarios and can be used on real Twitter data, where one does not know if the text and image labels match prior to classification but can perform classification of the tweet text and images using multimodal models.

4.3 Approach

In this section, we introduce some notations and elaborate the details of our approach. Suppose a tweet i consists of both text and image. We denote the tweet text by T_i , while the corresponding label of the text is denoted by yt_i . Similarly, we denote the tweet image by I_i , while the label of the image is denoted by yi_i . In our study, the labels of both text and images can take values *informative* or *non-informative*. For a tweet i which contains both

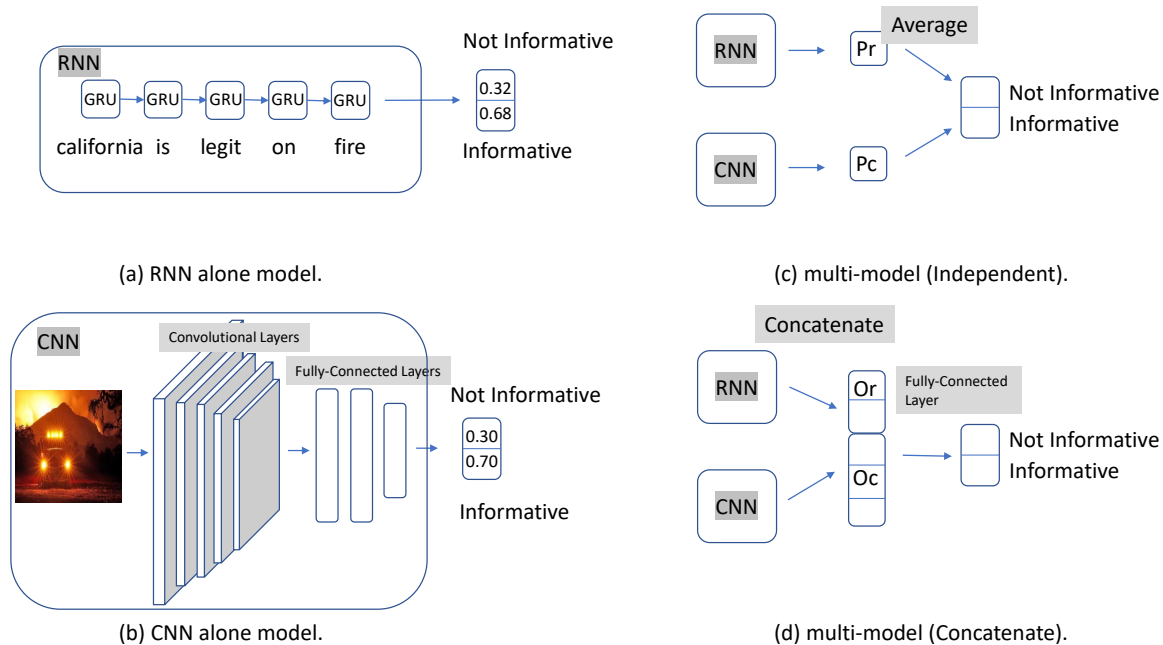


Figure 4.1: Architecture of proposed model

text and an image, the text label yt_i and the image label yi_i can be the same or different, as can be seen in the examples in Table 1. Therefore, the label of a tweet containing both text and an image can be assigned based on the original text labels, or based on the original image labels, unless the text and image have identical labels.

Regardless of the way tweet labels are assigned, a multi-modal labeled tweet instance is denoted by $\{T_i, I_i\}$, y_i for $i = 1, \dots, n$. Given a set of such multi-modal labeled tweet instances, the goal is to learn a model that uses both the text T_i and image I_i to predict the label y_i . The architecture of our model is shown in Figure 4.1. As can be seen, the model contains two distinct modules: a recurrent neural network (RNN) for processing the text and a convolutional neural network (CNN) for processing the image. Next, we describe the text model RNN, which predicts label y_i using T_i , and the image model CNN, which predict label y_i using I_i . Then, we describe how we combine the two models into a multi-modal model which predicts label y_i using both modalities of the tweet, $\{T_i, I_i\}$.

4.3.1 RNN

Recurrent Neural Networks (RNNs) have been used extensively in text classification [24]. An RNN learns sequential patterns by learning information from previous word together with information from the current word. There are many variants of RNN networks. In this study, we use Gated Recurrent Unit (GRU) networks [26]. The advantage of a GRU is that it can help capture long-term dependencies, while maintaining a relatively simple architecture. The text T_i consists of a sequence of words $\{w_1, \dots, w_j\}$, where j is the length of the text. As shown in Figure 4.1 (a), the information of each word (current GRU unit/state) will feed-forward to the next GRU unit. Formally, $g_i = GRU(w_i, g_{i-1})$. The output y of the network will be generated by the last word and the last GRU unit output. $\hat{y} = GRU(w_j, g_j)$. The model output is the prediction class: informative or non-informative.

4.3.2 CNN

Convolutional Neural Networks (CNNs) have been widely used in image classification. They learn spatial patterns in an image by extracting information from the surrounding area of each point through the means of a filter. As shown in Figure 4.1 (b), a CNN contains convolutional layers and fully-connected layers. The convolutional layer uses a sliding window on the image matrix to “walk” a filter through each image point. The filter identifies predictive information that may be repeated at different points in the image. A convolutional layer is usually followed by a pooling layer, which selects important information by taking the max or average in a window. After several convolutional layers, several fully-connected layers are used to produce final classification decisions. There are several CNN model architectures published. We used AlexNet [20] because it is relatively simple and efficient architecture. However, the CNN component of the model can potentially be improved by utilizing a deeper and more effective CNN architecture.

4.3.3 Multi-Modal Model

A multi-modal model is obtained by combining models corresponding to different modalities into one global model. In this paper, we combined the RNN and CNN models into one global model. As described above, the RNN component consumes text and the CNN component consumes images. The combined model will output one label. There are two ways in which we combine the RNN text and the CNN image models.

Train Text and Image Models Independently, and Average Predictions

The first way of combining the text and image models is to train the RNN and CNN models independently and then take the average of their prediction probabilities to get the final output of the network, as shown in Figure 4.1 (c).

$$Pr = RNN(T_i)$$

$$Pc = CNN(I_i)$$

$$\hat{y}_i = average(Pr, Pc)$$

where Pr/Pc represent the output probabilities from RNN/CNN. This idea is very straightforward and several works have already evaluated it in disaster management [89]. One disadvantage of this model is that it is hard to find tweets where the text and image labels match. Given that the two models are trained independently, their predictions cannot be combined for tweets where the text and image have different labels. Another disadvantage is that the AUC for this model will generally be lower than the best result because of the averaging operator.

Concatenate the Text and Image Representations and Co-train the Corresponding Models

Another way to combine the RNN and CNN models is to concatenate the outputs of their last layers, as shown in Figure 4.1 (d), and feed the combined representation to a fully

connected layer, which makes the final prediction.

$$Or = RNN(T_i); Oc = CNN(I_i)$$

$$\hat{y} = fc(concat(Or, Oc)) = ReLu(W * concat(Or, Oc) + b)$$

where Or/Oc represent the last layers of the RNN/CNN models, respectively, and fc is the fully-connected layer. W and b are the parameters in the fully-connected layer. The activation function is the ReLu function, where $f(x) = max(0, x)$. Fully-connected layers are widely used in neural networks, especially to connect the final representation/embedding layer to the classification layer. In this model, we extract representations from tweet text and image in the $concat(Or, Oc)$, and predict the label \hat{y} using the fully-connected layer. In this case, the two models will be co-trained together. The advantage of this way of combining the RNN and CNN is that the model will learn useful linking information automatically. If the image is not helpful to the text classification, this will also be presumably captured by the model. Thus, it is expected that the performance with respect to the base models will not be degraded.

4.4 Experimental Results

4.4.1 Dataset

The dataset we used in this study is the CrisisMMD dataset that was published by Alam et al. [3]. The dataset contains tweets crawled during seven disaster events. For each tweet, the text and image of the tweet were labeled as informative/non-informative. The labels for text and image might be different. We show the text label distribution in Table 4.2 left panel, and the label distribution for text labels matched with image labels in Table 4.2 right panel. As can be seen, the disaster related tweet datasets are quite unbalanced in terms of informative and not informative instances.

We also show the text/image label distribution in Table 4.3. We use the Hurricane Irma

Table 4.2: Statistics of the the dataset CrisisMMD

Before filtering. Original text labels.				After filtering. Matched labels.					
	Dataset	Not Informative	Informative	Total		Dataset	Not Informative	Informative	Total
D0	California Fire	345	1245	1590	D0	California Fire	282	923	1205
D1	Hurricane Harvey	1105	3334	4439	D1	Hurricane Harvey	906	2262	3168
D2	Hurricane Irma	957	3564	4521	D2	Hurricane Irma	767	2032	2799
D3	Hurricane Maria	1714	2844	4558	D3	Hurricane Maria	1295	1813	3108
D4	Iraq Iran Earthquake	104	493	597	D4	Iraq Iran Earthquake	102	398	500
D5	Mexico Earthquake	350	1030	1380	D5	Mexico Earthquake	315	806	1121
D6	Srilanka Floods	655	367	1022	D6	Srilanka Floods	632	229	861

Table 4.3: Hurricane Irma text/image distribution

txt/image	informative	non-informative
informative	2032	1532
non-informative	190	767

dataset as an example. The other datasets also show similar patterns. As can be seen in Table 4.3, if the image is informative, 90% (2032 out of 2222) of text is also informative; if the image is not informative, about 33% of text is not informative. This suggests that if the image is informative, the text is usually informative; if the image is not informative, the text might be informative or not.

4.4.2 Experimental Setup

Our multi-modal model aims to improve performance by learning from text and image together, so that the model will only output one decision for a text/image pair. We evaluate two scenarios in this study. The scenarios differs in how we filter the dataset and select labels for a text-image pair.

- In the first scenario, we filter out the tweets for which the corresponding text and image labels do not match. The data will be $\{T_i, I_i\}, y_i$ where $i = 1, \dots, n'$ and $n' < n$. The multi-model will predict the matched label y_i from $\{T_i, I_i\}$. We compare three results for this scenario: (1) Text only model with data T_i, y_i ; (2) Image only model with data I_i, y_i ; (3) multi-modal model with data $\{T_i, I_i\}, y_i$.
- In the second scenario, we predict text label using both text and image information,

and image label using both text and image information. The text label data will be $\{T_i, I_i\}$, yt_i where $i = 1, \dots, n$, while the image label data will be $\{T_i, I_i\}$, yi_i where $i = 1, \dots, n$. This scenario is more realistic, as in practice we don't know the text/image label in advance (i.e., we can't tell if they match or not), and thus we cannot filter out unmatched label tweets. We compare two results for this scenario for text labels: (1) Text only model; (2) multi-modal models. Similarly, we compare two results for image labels: (1) Image only model; (2) multi-modal models. While predicting text or image labels, the multi-model will learn some useful information from the other modality to complement the modality whose labels are predicted.

We evaluate the performance for the proposed model on the seven datasets from Crisis-MMD. Each dataset is randomly split into training (70%) and testing (30%) set. To avoid splitting bias, we run each experiment three times on each dataset and average the results over the three runs. We use Hurricane Maria to select hyper-parameters, given that this dataset is larger than others. Specifically, we split Hurricane Maria into training (50%) and development (20%) subsets and select hyper-parameters based on the development subset. Specifically, we use Adam Optimizer with learning rate as 0.001. To avoid over-fitting, we train each RNN model for at 10 epochs and each CNN model for 30 epochs.

4.4.3 Evaluation Metric

We use several standard evaluation metrics as defined below.

Accuracy

The accuracy represents the number of correctly predicted instances divided by the total number of samples:

$$Accuracy = \sum_i I_{\hat{y}_i=y_i}/n$$

where I is the indicator function, $I_{\hat{y}_i=y_i} = 1$ if $\hat{y}_i = y_i$; $I_{\hat{y}_i=y_i} = 0$ otherwise, and n is the sample size. The disadvantage of this metric is that it cannot capture well the performance

on unbalanced data. If the model classifies all samples into one class, the accuracy is still high but the model actually does not have good prediction ability.

AUC

We also use AUC (Area Under The Curve) to measure the models classification performance. As the name suggests, AUC is the area under a curve, specifically the curve which plots TPR vs. FPR at all possible thresholds, where True Positive Rate (TPR) is

$$TPR = \frac{TP}{TP + FN}$$

and False Positive Rate (FPR) is

$$FPR = \frac{FP}{FP + TN}.$$

This is a measure which balances the True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN), and better captures the performance for unbalanced data.

F1 Score

F1 Score is another measure which balances the precision and recall. In disaster management, the dataset is generally unbalanced. This might be a better measure than Accuracy to make the comparison.

4.4.4 Experimental Results

The experimental results for the first scenario are shown in Table 4.4. We compare the results for text alone, image alone and two types of multi-modal models. These results are performed on the matched label tweets for which the statistics are shown in Table 4.2, right panel. Because the datasets are unbalanced, we focus our discussion on the AUC and F1 scores. As can be seen, the RNN alone network has better AUC than the CNN alone

Table 4.4: Experimental results for the first scenario (image/text matched label results). D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods

		D0	D1	D2	D3	D4	D5	D6	Avg
Text only (RNN)	Acc	75.15	79.86	78.70	76.20	74.25	72.73	90.67	78.22
	AUC	71.51	83.47	81.38	84.33	66.65	72.90	92.21	78.92
	F1	60.45	73.96	70.92	74.81	55.63	64.57	87.68	69.71
Image only (CNN)	Acc	76.68	76.60	78.65	66.75	74.80	74.19	80.48	75.45
	AUC	68.72	77.93	62.41	69.46	70.53	69.03	85.55	71.95
	F1	75.03	75.40	71.54	66.41	73.18	71.57	80.27	73.34
Multimodal (RNN+ CNN) (independent+average)	Acc	78.50	79.32	74.98	74.23	76.15	76.27	84.60	77.72
	AUC	71.93	84.22	75.53	82.37	70.25	79.38	86.04	78.53
	F1	76.04	78.17	72.26	73.89	71.89	73.27	83.90	75.63
Multimodal (RNN+CNN) (concat+co-train)	Acc	75.45	79.78	79.91	76.67	75.06	73.62	91.59	78.87
	AUC	71.49	84.29	81.36	83.64	71.64	76.71	96.50	80.80
	F1	73.20	79.20	79.07	76.59	74.13	72.92	91.85	78.13

network, while the CNN alone network has better F1-score. However, the combination of the two networks has better performance overall. As can be seen, the average performance of the multimodal model over all experiments/datasets has better results than both the CNN model and RNN model. This shows that the combination of text and image information leads to better classification labels. The concatenation+co-training multimodal model gives the best results overall. By looking at specific datasets, we can see that the multimodal model has better results for hurricane disasters (D1, D2, D3), as compared to the results for the fire disaster (D0). The reason for better performance might be the fact that the hurricane datasets are larger and enable better learning of the models. Based on the results in Table 4, we can conclude that the proposed multimodal model can improve the classification performance for the matched label scenario. However, this scenario is not very useful from a practical point of view, as we don't know if the text and image labels will match when we try to use this model for an ongoing disaster.

The experimental results for the second scenario are shown in Table 4.5 and Table 4.6 for text labels and image labels, respectively. As can be seen in Table 4.5, for text labels, the average multimodal model has better overall F1-score than the RNN alone model. However, the AUC for the average multimodal model is overall worse than that of the RNN alone

Table 4.5: Text label results. D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods

		D0	D1	D2	D3	D4	D5	D6	Avg
Text only (RNN)	Acc	75.90	79.17	78.81	70.41	76.23	74.90	89.84	77.89
	AUC	66.08	77.60	73.60	75.74	70.47	71.17	96.07	75.82
	F1	57.82	68.33	62.19	66.83	55.64	65.23	90.19	66.60
Multimodal (RNN+ CNN) (independent+average)	Acc	77.62	76.34	76.15	67.11	81.11	74.91	78.26	75.93
	AUC	67.35	71.65	63.49	69.47	67.98	73.11	83.47	70.93
	F1	74.29	73.48	72.61	65.05	77.04	71.47	78.44	73.20
Multimodal (RNN+ CNN) (concatenate+co-train)	Acc	73.65	77.86	79.69	71.56	80.89	75.35	89.86	78.41
	AUC	68.59	77.79	73.90	76.81	72.28	72.95	95.99	76.90
	F1	71.45	76.42	77.18	70.72	79.57	74.77	89.97	77.15

Table 4.6: Image label results. D0: california wildfires D1: hurricane harvey D2: hurricane irma D3: hurricane maria D4: iraq iran earthquake D5: mexico earthquake D6: srilanka floods




		D0	D1	D2	D3	D4	D5	D6	Avg
Image only (CNN)	Acc	76.68	76.83	78.65	64.33	83.33	74.19	81.20	76.46
	AUC	68.72	70.79	62.41	65.11	76.30	69.03	85.47	71.72
	F1	75.03	73.14	71.54	60.18	80.43	71.57	80.93	73.26
Multimodal (RNN+ CNN) (independent+average)	Acc	77.13	77.08	77.74	65.58	81.33	75.81	78.40	76.15
	AUC	67.41	71.96	65.30	66.40	70.13	72.31	84.76	71.18
	F1	73.92	73.15	73.75	62.28	75.25	72.00	78.27	72.66
Multimodal (RNN+ CNN) (concatenate+co-train)	Acc	75.11	78.42	80.38	71.42	80.00	73.39	92.00	78.67
	AUC	70.11	77.92	72.24	77.35	72.28	75.32	96.09	77.33
	F1	74.09	77.68	78.38	69.63	78.01	73.34	92.06	77.60

model. The concatenation+co-train multimodal model has AUC performance better than that of the RNN alone model, and has much better performance in terms of the F1 Score. The concatenation+co-train multimodal model also has better performance than the average multimodal model overall. Considering specific datasets, the multimodal model has better performance on hurricane datasets, earthquake datasets and comparable results on flood datasets. As in the first scenario, this shows that our proposed model benefits from larger datasets. Based on these results, our conclusion is that the multimodal model (based on concatenation+co-training) has better performance than the text only model. Thus, we suggest to use the multimodal model when predicting the labels of tweet texts.

Similar patterns can be seen in Table 4.6 for image labels. When using text to supplement image information, the results are comparable between the image only model and the average multi-modal model. The concatenation+co-train multimodal model has better performance overall. Similar individual pattern can be seen here. The multimodal model benefits most from the larger hurricane datasets. The results suggests that text can also help the prediction of image. Therefore, the concatenation+co-train multimodal model can be used in image classification.

We also show some text prediction examples in Table 4.7 to get insights into how the multimodal model helps. In the first two examples, the text is informative but the RNN model misclassifies it as not-informative. The corresponding images show a damaged church, and a hurricane, respectively, and help the multimodal model assign the correct labels. The text in the third example is somewhat confusing cannot be used to predict the tweet as not informative. However, the corresponding image is clearly not informative with respect to a disaster and helps the multimodal model correctly classify the tweet. These examples support our conclusion that the information from an image can help the prediction of text labels.

Table 4.7: Text Classification Examples: the multimodal model helps improve the classification

Text Label	Informative	Informative	Not Informative
Text	Baptist church damaged by #harvey can't use sanctuary	A piece of a #Harvey rain band drifts north through Breton Sound	Hurricane Harvey Relief Supply Drive...and efforts towards IRMA
Image Label	Informative	Informative	Not Informative
Image			
RNN prediction	Not Informative	Not Informative	Informative
Multimodal prediction	Informative	Informative	Not Informative

4.5 Conclusions

In this paper, we propose a multi-modal model which combines text and image information through the means of RNN and CNN models. We evaluate different scenarios and identify a scenario where it is useful to use the multi-modal model as opposed to specific modality models. Specifically, we show that our proposed multimodal model can be used to improve the text classification by adding the image information and improve the image classification by adding the text information. The future work for this study will be focused on exploring different text/image classification models to improve the multi-modal model architecture. We will evaluate our approach on disaster-specific data (e.g., learn a model for hurricanes, another model for earthquakes, etc.) and also on cross-disaster data (e.g., learn a model on hurricanes and use it on a new type of disaster), so that we can get more insights into the model performance. We will also evaluate our proposed models on other datasets, including

multi-class datasets for predicting situational awareness categories.

Chapter 5

Localizing and Quantifying Damage in Social Media Images

5.1 Introduction

Fast detection of damaged areas after an emergency event can inform responders and aid agencies, support logistics involved in relief operations, accelerate real-time response, and guide the allocation of resources. Most of the existing studies on detecting and assessing disaster damage rely heavily on macro-level images, such as remote sensing imageries [96, 97], [98], or imageries transmitted by unmanned aerial vehicles [99]. Collection and analysis of macro-level images require costly resources, including expensive equipment, complex data processing tools, and also good weather conditions. To benefit the response teams, the macro-level images have to be collected and analyzed very fast, which is not always possible with traditional collection and analysis methods.

With the growth of social media platforms in recent years, real-time disaster-related information is readily available, in the form of network activity (e.g., number of active users, number of messages posted), text (e.g., tweets), and images posted by eyewitnesses of disasters on platforms such as Twitter, Facebook, Instagram or Flickr. Many studies have shown the utility of social media information for disaster management and response teams.

For example, the analysis of text data (e.g., tweets from Twitter) has received significant attention in recent works [100], [80], [48], [64], [101]. However, social media images, while very informative [58], have not been extensively used to aid disaster response, primarily due to the complexity of information extraction from (noisy) images, as compared to information extraction from text.

By contrast with macro-level images, social media images have higher “resolution”, in the sense that they can provide detailed on-site information from the perspective of the eyewitnesses of the disaster [58]. Thus, social media images can serve as an ancillary yet rich source of visual information in disaster damage assessment. Pioneering works with focus on the utility of social media images in disaster response include [86], [18], where the goal is to use convolutional neural networks (CNN) to assess the severity of the damage (specifically, to classify social media images based on the degree of the damage as: *severe*, *mild*, and *none*).

Due to ground-breaking developments in computer vision, many image analysis tasks have become possible. Disaster management and response teams can benefit from novel image analyses that can produce quantitative assessments of damage, and inform the relief operations with respect to priority areas. In this context, it would be useful to locate damage areas in social media images (when images contain damage), and subsequently use the identified damage areas to assess the damage severity on a continuous scale. Possible approaches for localizing damage in social media images include object detection [102, 103] and image segmentation [104]. Object detection can be conducted by classifying some specific regions in an image as containing damage or not. For example, Cha et al. [102] used a convolution neural network (CNN) to classify small image regions (with 256×256 pixel resolutions) as containing concrete crack damage or not. Maeda et al. [103] used a state-of-the-art object detection approach, called Single Shot MultiBox Detector (SSD) [105], to detect several types of road damage. Image segmentation has been used in [104] to detect building damage based on high resolution aerial images.

Regardless of the method used, object detection or image segmentation, existing approaches for localizing damage first identify objects (i.e., potential damage regions) and sub-

sequently classify the objects as *damage* (sometimes, *severe* or *mild*) or *no damage*. Thus, there is a conceptual mismatch in the way existing approaches are used, given that damage is generally regarded as a high-level concept rather than a well-defined object. By first identifying objects and then assigning discrete hard-labels to them, existing approaches produce a clear-cut boundary for the damaged areas, although a smooth boundary would be more appropriate.

Contributions: Inspired by the technique called Class Activation Mapping (CAM) [42], we propose a novel approach, called Damage Detection Map (DDM), to generate a smooth damage heatmap for an image. Our approach adopts the gradient-weighted CAM (Grad-CAM) [106] technique to localize the area in an image which contributes to the damage class. Based on the damage heatmap, we also propose a new quantitative measure, called Damage Assessment Value (DAV), to quantify the severity of damage on a continuous scale. One advantage of the proposed approach is that it only requires annotators to label images as having damage or no damage, as opposed to requiring annotators to localize the damage. Thus, our approach makes the disaster damage localization possible, and extends the use of social media images in disaster assessment. We also evaluate the usefulness of the proposed approach for localizing and quantifying different types of infrastructure damage, including building damage, bridge damage and road damage. Furthermore, in addition to Grad-CAM, we also use its Grad-CAM++ extension [43], which can identify multiple occurrences of a class object, and compare Grad-CAM and Grad-CAM++ in the context of damage localization and quantification. Extensive experimental results show that our approach makes the disaster infrastructure damage localization possible, and thus extends the use of social media images in disaster assessment.

The rest of this paper is organized as follows: we describe the proposed approaches for generating DDM heatmaps, and computing DAV scores in Section 5.2. We describe the experimental setup and results in Section 5.3. We discuss related work in Section 5.5, and conclude the paper in Section 5.6.

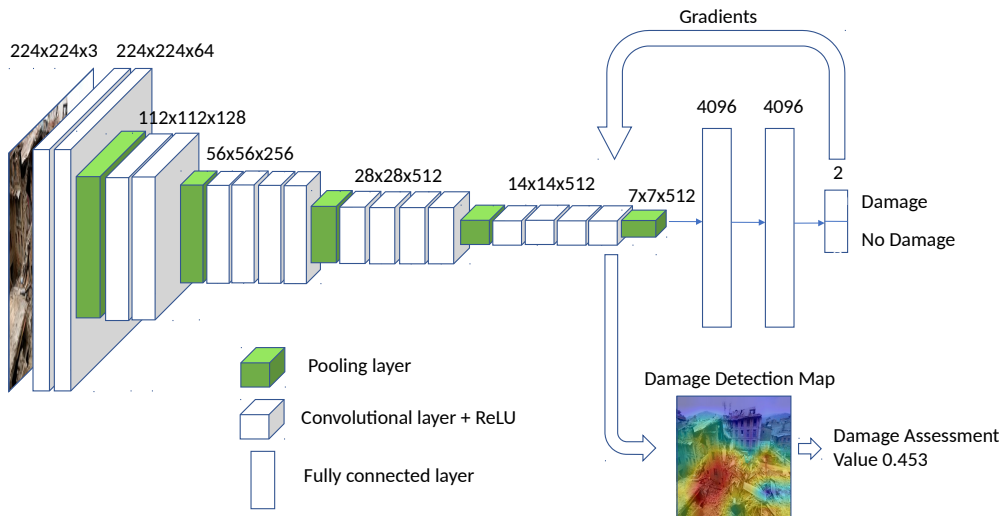


Figure 5.1: Overview of the proposed approach.

5.2 Proposed Approach

Our approach generates a Damage Detection Map, which visualizes the damage area for a given image, and a score DAV, which quantifies the severity of the damage. The main components of our approach, shown in Fig. 5.1, are the following: 1) a CNN that classifies images into two classes, *damage* or *no damage*; 2) a class activation mapping, which generates the DDM map by weighting the last convolutional layer of the CNN model; 3) finally, the damage severity score computed by averaging the values in the map. The details for the three components of our approach are provided in what follows.

5.2.1 Convolutional Neural Networks

Convolutional neural networks [107] have been used successfully for many image analysis tasks [9, 8]. The ImageNet annual competition (where a dataset with 1.2 million images in 1000 categories is provided to participants) has led to several popular architectures, including AlexNet [20], VGG19 [21], ResNet [22] and Inception [108]. We choose VGG19 as the architecture for our CNN model, as VGG19 has good classification accuracy and it is relatively simpler compared to ResNet and Inception. Furthermore, the pre-trained model

are available for VGG19 and it is easy to fine tune them for different classification problems.

VGG19 [21] contains 16 convolutional layers (with 5 pooling layers) and 3 fully connected layers. Each convolution layer is equipped with a non-linear ReLU activation [20]. The convolutional layers can be seen as feature extraction layers, where each successive layer detects predictive image features (i.e., image fragments that correspond to edges, corners, textures, etc.) at a more abstract level than the previous layer.

As can be seen in Fig. 5.1, the size of the input to the convolutional layers is reduced by a factor of 2 through max-pooling layers, but not all convolution layers are followed by a max-pooling layer. After every max-pooling layer, the width of the convolution layer (i.e., number of filters used) increases by a factor of 2. After the last max-pooling layer, there are two fully connected layers with dimension 4096, and another fully connected layer whose neurons correspond to the categories to be assigned to the input image. The last layer of the standard VGG19 model has dimension 1000 because VGG19 was originally trained on a dataset with 1000 categories. However, as we are interested in using VGG19 to classify images in two categories, *damage* and *no damage*, we change the dimension of the last fully connected layer from 1000 to 2. Overall, the model includes more than 130 million parameters, and it takes a significant amount of time (and a large number of images) to train it accurately [21]. However, the model parameters are highly transferable to other image classification problems [109]. Thus, to avoid the need for a large number of images, we initialize our model with the pre-trained VGG19 model (except for the last fully connected layer), and fine tune it using disaster-related images. More specifically, given a training image x with label y represented as a one-hot vector (e.g., if the label is *damage*, then $y = [1, 0]$, otherwise $y = [0, 1]$), all the parameters θ will be updated by:

$$\theta \leftarrow \theta - \mu \frac{\partial \mathcal{L}(y, \text{CNN}(x))}{\partial \theta}, \tag{5.1}$$

where μ is learning rate, \mathcal{L} is the cross-entropy loss, and $\text{CNN}(x)$ is the output of the CNN given input x .

5.2.2 Damage Detection Map

Our proposed Damage Detection Map (DDM) is inspired by the Gradient-weighted Class Activation Mapping (GCAM) [106]. In a general classification problem, for an input image and a trained CNN model, GCAM makes use of the gradients of a target category to compute a category-specific weight for each feature map of a convolution layer. The weights are used to aggregate the feature maps of the final convolutional layer, under the assumption that the last level captures the best trade-off between high-level semantic features and spatial information. The resulting maps can be used to identify the discriminative regions for the target category (which explain the CNN model’s prediction), and implicitly to localize the category in the input image. Thus, GCAM can be seen as a weakly supervised approach, which can localize a category in an image based only on global image labels [106]. Furthermore, the GCAM localized categories or objects (shown using heatmaps) have soft boundaries, and can be used to gain both insight and trust into the model. This makes GCAM particularly attractive for the problem of localizing damage in disaster images, assuming that only coarse labeling of images as *damage* or *not damage* is available for training. The heatmaps showing categories of interest using soft-boundaries are very appropriate for localizing damage, as damage boundaries are inherently soft. Moreover, the heatmaps that explain the model’s predictions can be used to gain the trust of disaster management teams, and thus increase the usability of social media images in disaster response and recovery.

Given the above introduction to GCAM and motivation for use in the context of disaster damage localization, we formally describe the GCAM approach [106] in remaining of this subsection. Consider a 14×14 matrix S , such that

$$s_{i,j} = \text{ReLU}\left(\sum_k w_k f_{(i,j)}^k\right) \quad (5.2)$$

where w_k is a gradient-based weight parameter (defined in Equation 5.3) corresponding to each feature map f^k in the last convolutional layer (of dimension $14 \times 14 \times 512$), and $f_{(i,j)}^k$ represents the value at location (i, j) in the k -th feature map, for $i = 1, \dots, 14$, $j = 1, \dots, 14$,

$k = 1, \dots, 512$. The ReLU function is used to cancel the effect of the negative values, while emphasizing the effect of the positive values. Given an input image x and the output of the CNN for the damage class y_D (just before the softmax function is applied), the weights w_k are determined as the sum of the gradients of output y_D with respect to $f_{(i,j)}^k$, for all i, j . Specifically:

$$w_k = \frac{1}{14 \times 14} \sum_{i,j} \frac{\partial y_D}{\partial f_{(i,j)}^k} \quad (5.3)$$

The feature maps f^k in Equations (5.3) and (5.2) are in the last convolutional layer, as this layer generally shows a good trade-off between high-level features and spatial information in the original image. Our final goal is to localize damage in disaster images, in other words to find regions that are discriminative for the damage class. Intuitively, the gradient with respect to a neuron in the final convolutional layer represents the contribution of the neuron to the class label. The procedure described above produces a 14×14 matrix S (and correspondingly a 14×14 image). Using an interpolation technique, we further resize S to S_C to match the original dimensions of the image. The heatmap showing the S_C values is the final Damage Detection Map.

Chattopadhyay et al. [43] identify two main drawbacks of the Grad-Cam approach, and proposed an extension, called Grad-CAM++, to address the Grad-CAM drawbacks. First, they observed that Grad-CAM does not properly identify/localize all occurrences of a class object. Furthermore, Grad-CAM may not always localize the whole class object, but only parts of it. Chattopadhyay et al. [43] noted that if a target class object has multiple occurrences in an image, marked by footprints with different orientations and sizes, then only the object occurrences with larger footprint will be visible in the heatmaps produced by Grad-CAM. To alleviate this problem, they proposed Grad-CAM++, which assigns weighting coefficients $\alpha_{i,j}^k$ to the pixel (i, j) gradients for target class y and feature map k . Thus, in Grad-CAM++ Equation (5.3) becomes:

$$w'_k = \sum_{i,j} \alpha_{i,j}^k \text{ReLU} \left(\frac{\partial y}{\partial f_{(i,j)}^k} \right) \quad (5.4)$$

The effect of the ReLU function in Equation (5.4) is similar to the effect of ReLU in Equation (5.2): it helps emphasize the positive values that contribute to the importance (i.e., weight) of a particular feature map. Chattopadhyay et al. [43] derive closed-form formulas for calculating the pixel-wise coefficients $\alpha_{i,j}^k$ for a feature map f_k and a target class y . Replacing w_k with w'_k in Equation (5.2) leads to higher weights for features maps corresponding to target objects with a smaller footprint in the original images, and thus allows for multiple occurrences of the object to be identified (and similarly allows for the localization of an object in its entirety).

We will experiment with both Grad-CAM and Grad-CAM++ approaches to identify and localize damage in disaster images.

5.2.3 Measuring the damage severity

When a disaster occurs, eyewitnesses of the disaster will produce a huge number of images in a short period of time. An important goal of disaster assessment is to extract and concisely summarize the information contained in the images posted by eyewitnesses. To meet this demand, we propose to use a damage assessment value (DAV) derived from the DDM heatmap to represent the damage severity for each image.

The disaster damage map uses numerical values to measure the intensity of each pixel of the image (the higher the intensity, the more severe the damage), and can be represented as a heatmap. We take the average over all the numerical values in the heatmap of a given image, and use the resulting value as an overall score for the severity of the damage. Formally, we define the damage assessment value (DAV) as:

$$DAV = \frac{1}{14 \times 14} \sum_{i,j} s_{i,j} \quad (5.5)$$

where $s_{i,j}$ are the elements of the S matrix defined in Equation (5.2), and 14×14 is the dimension of the matrix S .

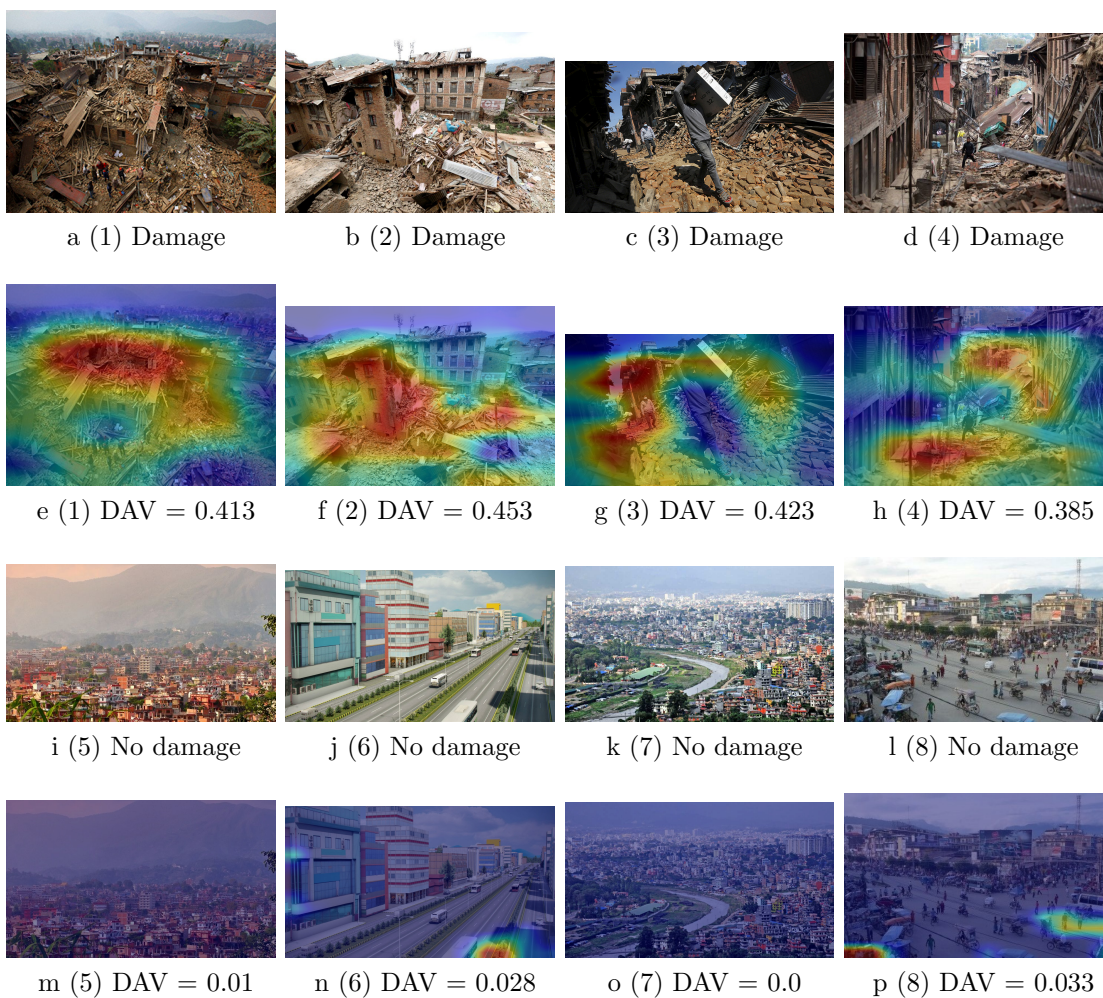


Figure 5.2: Examples of images in our dataset, and their corresponding DDM heatmaps and DAV scores. The first row shows examples of images in the *damage* class, followed by their corresponding DDM heatmaps and the DAV scores. Similarly, the third row shows examples of images in the *no damage* class, followed by their corresponding DDM heatmaps and DAV scores.

Table 5.1: Social media image dataset consisting of images from four disaster events. Images are labeled as *Severe*, *Mild*, or *None*. The number of images in each class, and the total number of images for each disaster are shown.

Disaster	Severe	Mild	None	Total
Nepal Earthquake	5303	1767	11226	18296
Ecuador Earthquake	785	83	886	1754
Ruby Typhoon	76	325	463	864
Matthew Hurricane	97	89	130	316

5.3 Experimental Results I

We perform a series of experiments to evaluate the performance of our proposed method. The experiments are designed to answer the following questions: (i) can the Damage Detection Map accurately locate the damage areas, (ii) can the DAV score provide a reliable measure for damage severity.

5.3.1 Experimental Setting

Datasets

We used two datasets in our experiments. The first dataset is assembled using Google image search engine. Specifically, we performed two searches: first, we used ‘nepal’, ‘building’ and ‘damage’ as keywords, and crawled 308 *damage* images from the result; second, we used ‘nepal’ and ‘city’ as keywords, and crawled 311 *no damage* images. The keyword ‘nepal’ was used in both searches to ensure that the two sets of images have similar scene, while the keywords ‘building damage’ and ‘city’ were used to bias the search towards *damage* and *no damage* images, respectively. We show some sample *damage* and *no damage* images from the Google dataset in Fig. 5.2 in the first and third rows, respectively.

The second dataset used in our evaluation was previously used in [18], and consists of social media images posted during four different disaster events: Nepal Earthquake, Ecuador Earthquake, Ruby Typhoon, and Matthew Hurricane. For each disaster, the dataset contains images in three categories, representing three levels of damage: severe, mild, and none. Table 5.1 provides the class distribution for each disaster dataset.

In our experiments, each dataset is randomly split into a training set (80%) and a test set (20%). We will use the proposed approach to learn a CNN that discriminates between *damage* (including *severe* and *mild*) and *no damage* images, while also localizing the damage and identifying image features discriminative for the *damage* class.

Hyper-parameters

We used TensorFlow’s GradientDescentOptimizer to train the model using mini-batch gradient descent on a GeForce GTX 1070 graphic card. Based on preliminary experimentation with the Ecuador Earthquake and Ruby Typhoon datasets, we chose to use a learning rate of 0.001 and a batch size of 32 images in all our experiments. Furthermore, we used the dropout technique with a rate of 0.5 to prevent overfitting. The code for the VGG19 model was adapted from <https://github.com/machrisaa/tensorflow-vgg>.

5.3.2 Damage Detection Map Evaluation

We first use the Google dataset, where the *damage* images are easier to discriminate from the *no damage* images, to evaluate the Damage Detection Map (DDM) approach (intuitively the better the CNN classifier, the better the DDM map). Specifically, we train a CNN model on the Google dataset as described in Section 5.2.1. The training accuracy of the CNN model is 100%, and the test accuracy is 95.5%. Subsequently, we compute the DDM heatmaps as described in Section 5.2.2. The DDM heatmaps corresponding to the sample *damage* and *no damage* images are shown in Fig. 5.2, in the second and fourth rows, respectively. As can be seen, the regions with high intensity DDM generally correspond to damage.

To answer our first research question, specifically to understand if the DDM can accurately locate the damage areas, we will evaluate the localization capability of DDM by using the Intersection-Over-Union (IOU) measure, which is frequently used to evaluate object detection techniques [110]. The IOU measure is defined as follows:

$$\text{IOU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{5.6}$$

where the ‘Area of Overlap’ and ‘Area of Union’ are computed with respect to a ground truth image, where damage is manually marked. IOU takes values in $[0, 1]$. If the IOU value for a pair of GCAM damage marked image and the corresponding ground truth image is large, then the marked damage areas in the two images are similar, and thus the GCAM approach performs well in terms of damage localization.

We randomly select 10 images from the Google test dataset for manual annotation, and use the annotations to evaluate damage localization using IOU scores. The scores are computed by comparing the automatically localized damage with the manually marked damage. As damage is not an object, heatmaps with smooth boundaries are preferable to bounding boxes when localizing damage. However, human annotators cannot provide precise heatmaps, unless they have professional knowledge of disaster damage, in which case their annotation would be very expensive. To reduce the cost, generally human annotators will simply mark the regions of an image that contain damage (resulting in a binary included/not included representation). We used the tool LabelMe (<https://github.com/wkentaro/labelme>) to mark the damage.

To compare heatmaps with images marked by annotators in terms of IOU values, we transform the heatmaps to a binary representation as follows: we determine the maximum value in S_C and use 20% of the maximum value as a cutoff value for including a region in the disaster damage “object” or not [42]. In other words, only the regions in DDM with values larger than the cutoff value will be part of the localized damage. In the resulting transformed image (as well as in the human annotated images), the damage pixels have value 255, while no damage pixels have value 0. We show the result of the IOU evaluation on the sample consisting of 10 Google images, by comparison with annotations from two independent annotators A and B, in Table 5.2. To understand the difficulty of the task of identifying damage in a picture, we also compute IOU values that show the agreement between annotators A and B.

Conventionally, if the IOU value corresponding to a detected object (marked with a bounding box) is larger than 0.5, the detection/localization of that object is considered to be correct [110]. However, given that the damage is not an object but a concept, we find that

Table 5.2: IOU for 10 Google images. Annotation A and Annotation B are provided independently by two annotators. The image heatmaps were transformed to a binary representation by using a threshold equal to 20% of the max value in the DDM

	Image	1	2	3	4	5	6	7	8	9	10	Average
IOU	Annotation A versus DDM	0.334	0.401	0.568	0.360	0.474	0.270	0.349	0.156	0.418	0.477	0.380 \pm 0.116
	Annotation B versus DDM	0.444	0.623	0.633	0.473	0.583	0.445	0.258	0.440	0.616	0.658	0.517 \pm 0.126
	Annotation A versus B	0.677	0.541	0.744	0.681	0.695	0.546	0.673	0.237	0.621	0.689	0.610 \pm 0.146

the 0.5 threshold is too strict in the context of detecting and localizing damage, especially as the average IOU agreement between annotators is 0.610. If we consider an IOU score of 0.4 as detection, then the GCAM-based approach detects 50% of the damage marked by annotator A, and 90% of the damage marked by annotator B. To better understand the results, in Fig. 5.3, we show the annotations for three sample images, specifically, image 5 for which the IOU agreement with both annotators is above 0.4, image 7 for which the IOU agreement with both annotators is smaller than 0.4, and image 1, for which the agreement with Annotator A is below 0.4 and the agreement with the Annotator B is above 0.4. As can be seen, the damage areas marked based on the DDM heatmaps look accurate overall, and in some cases better than the damage areas marked by the annotators, which confirms that identifying damage is a subjective task, and that DDM can be useful in localizing damage and providing visualizations that can help responders gain trust in the annotations produced by deep learning approaches.

5.3.3 Damage Assessment Value Evaluation

We calculate the Damage Assessment Value (DAV) as described in Section 5.2.3. The DAV values for the sample Google images in Fig. 5.2 are shown below the corresponding heatmap images. As can be seen, images that present a more severe damage scene have higher DAV values, while images with no damage have DAV values very close to zero.

We also use the disaster datasets shown in Table 5.1 to further evaluate the DAV values. For each disaster, we use the corresponding training set to fine-tune the CNN model to that specific disaster. For this purpose, we combined the original *severe* and *mild* damage classes into a single *damage* class, while the original *none* class label is used as *no damage* class. We

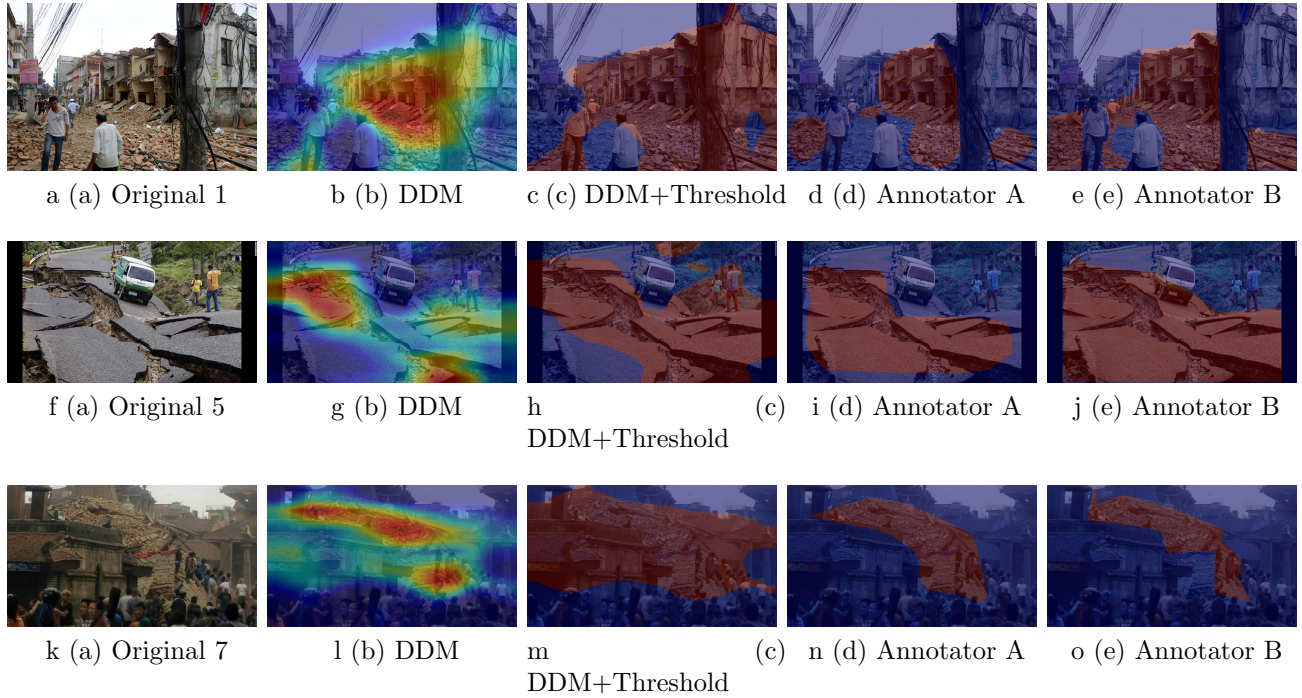


Figure 5.3: Damage Detection Map (DDM) versus human annotations for images 1, 5 and 7 in Table 5.2. (a) Original Google image; (b) DDM damage heatmap; (c) Binary representation of the DDM computed with a threshold whose value is 20% of the max value in the DDM heatmap; (d) Damage marked by Annotator A; (e) Damage marked by Annotator B.

used the CNN model of each disaster to produce DDM heatmaps for all test images from that disaster. Subsequently, we used the heatmaps to generate DAV values for each test image, and generated DAV density plots for the *damage* versus *no damage* classes, shown in Fig. 5.5. As can be seen in the figure, the density plots show clearly differentiable patterns between the two classes, with the *damage* class exhibiting higher values overall, as expected. However, there is also overlap in terms of DAV values for *damage* and *no damage* images. This can be partly explained by the noisiness of the dataset, and the difficulty of the task of separating images in damage severity classes, as also noted in [18]. To illustrate this claim, in Fig. 5.4, we show several images that seem to be mislabeled in the original dataset.



Figure 5.4: Examples of images mislabeled in the disaster dataset. The original image label and the DAV score are shown.

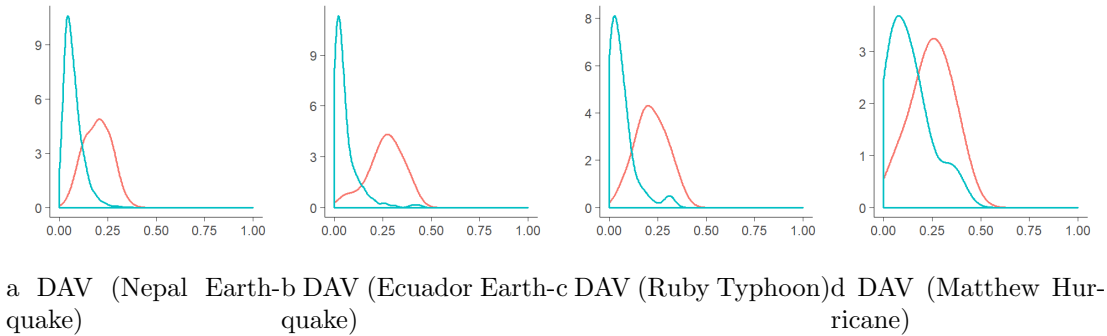


Figure 5.5: Smoothed density curves for DAV values. The *no damage* class is shown in blue and the *damage* class is shown in red. The graphs are based on the test set of each disaster dataset.

5.3.4 Classification using DAV values

In this section, we study the usefulness of the DAV values in classifying images into several damage categories. Specifically, we consider the *severe*, *mild*, and *none* categories, as the images in the disasters used in our study are already labeled with these categories. Using the training data, we perform a grid-search to find two threshold values for DAV, denoted by c_1 and c_2 , which minimize the classification error on the training data. Using these thresholds, we design a simple classifier as follows: test images with DAV values smaller than c_1 will be classified as *none*, those with DAV values in between c_1 and c_2 will be classified as *mild*, and finally those with DAV values greater than c_2 will be classified as *severe*. The classification results of our simple classifier (averaged over five independent runs), together with the average results of a three-class CNN model (trained on the corresponding training set of each disaster in each run), are reported in Table 5.3. The classification results of the simple classifier based on DAV are similar to those of the CNN model for Ecuador Earthquake

Table 5.3: Classification accuracy for each disaster dataset. The first row shows the average accuracy of our simple DAV-based classifiers (over 5 independent runs), together with standard deviation. The second row shows the average accuracy of the three-class CNN models. The third row shows the results published in [4], which used the same datasets.

Method	Nepal Earthquake	Ecuador Earthquake	Ruby Typhoon	Matthew Hurricane
DAV	0.801±0.013	0.886±0.018	0.793±0.017	0.533±0.043
VGG19	0.851±0.003	0.901±0.013	0.852±0.018	0.603±0.151
VGG16 [4]	0.840	0.870	0.810	0.740

and Matthew Hurricane (i.e., the results are not statistically different based on a t-test with $p \leq 0.05$). While the CNN accuracy looks better overall, the results indicate that DAV can capture severity damage (on a continuous scale), and it can help produce simple and interpretable classifiers.

5.4 Experimental Results II

1

5.4.1 Experimental Setup

We perform a series of experiments to evaluate the performance of our proposed method. The experiments are designed to answer the following questions:

1. Can the Damage Detection Map accurately locate the damage areas?
2. What type of damage can be localized more accurately?
3. Can the DAV scores provide a reliable measure for damage severity?
4. Among two-class (*damage* or *no-damage*), four-class (*building-damage*, *bridge-damage*, *road-damage*, or *no-damage*) or six-class (*building-damage*, *no-building-damage*, *bridge-*

¹This is part of a post-peer-review, pre-copyedit version of an article published in Social Network Analysis and Mining. The final authenticated version is available online at: <http://dx.doi.org/10.1007/s13278-019-0588-4>

damage, *no-bridge-damage*, *road-damage*, or *no-road-damage*) models, what model leads to better results overall?

5. Between Grad-CAM and Grad-CAM++, which approach is better?

Data Description

The main dataset we used in this paper was originally published in [18] and it is available from <http://crisisnlp.qcri.org>. The dataset was assembled from Google by using search queries such as *building-damage*, *bridge-damage*, *road-damage*. In addition to images in one of these three categories, the dataset contains images labeled as *no-damage*. The numbers of images in the *building-damage*, *bridge-damage*, *road-damage* and *no-damage* categories are 903, 813, 826, 463, respectively. From the original dataset, we manually filtered out images that did not contain buildings, bridges or roads, and also noisy/mis-labeled images. Furthermore, we manually classified the remaining *no-damage* images as *no-building-damage*, *no-bridge-damage* and *no-road-damage*, and used Google search to crawl more images in these specific no-damage categories. Finally, as we aim to localize damage in images, we manually marked the damage area in each damage image. As damage is not an object, heatmaps with smooth boundaries are preferable to bounding boxes when localizing damage. However, human annotators cannot provide precise heatmaps, unless they have professional knowledge of disaster damage, in which case their annotation would be very expensive. To reduce the cost, generally human annotators will simply mark the regions of an image that contain damage (resulting in a binary included/not included representation). We used the tool LabelMe, available from <http://labelme.csail.mit.edu>, to mark the damage. The statistics about the final dataset used are shown in Table 5.4. Sample images in the damage categories, together with the ground-truth localization annotations, are shown in Fig. 5.6. We will refer to this dataset as *Google dataset* in what follows.

We randomly split the Google dataset into training (80%) and test (20%) subsets. The VGG19 models were fine-tuned on the training subset, and the evaluation was performed on the test subset.

Category	Bridge	Building	Road	Total
Damage	347	709	525	1581
No-Damage	731	510	497	1738
Total	1078	1219	1022	3319

Table 5.4: Statistics about the Google damage dataset, which contains images in the following categories: *bridge-damage*, *no-bridge-damage*, *building-damage*, *no-building-damage*, *road-damage*, and *no-road-damage*

Severity	0	0.25	0.5	0.75	1	Total
Damage	46	15	47	86	56	250

Table 5.5: Statistics about the Building Damage Severity dataset, which contains building images in the following categories: *no-damage* (0), *slight-damage* (0.25), *moderate-damage* (0.5), *heavy-damage* (0.75), and *total-destruction* (1)

To evaluate the ability of the DAV scores to capture the severity of the damage in an image, we used the dataset published in [111]. This dataset contains images with building damage caused by natural disasters. Each image was manually labeled using one of the following categories, which are indicative of the degree of damage: *no-damage*, *slight-damage*, *moderate-damage*, *heavy-damage*, and *total-destruction*. Each category is associated with a numeric value: *no-damage* \leftrightarrow 0, *slight-damage* \leftrightarrow 0.25, *moderate-damage* \leftrightarrow 0.5, *heavy-damage* \leftrightarrow 0.75, *total-destruction* \leftrightarrow 1. The statistics about the final dataset used are shown in Table 5.5. Sample images with different degrees of severity, are shown in Fig. 5.7. We will refer to this dataset as *Building Damage Severity* dataset in what follows.

VGG19 Models Trained

We trained two-class, four-class and six-class VGG19 models. There are three two-class models trained to discriminate between *damage* and *no-damage* images, specifically, *building-damage* versus *no-building-damage*, *bridge-damage* versus *no-bridge-damage*, and *road-damage* versus *no-rod-damage*, respectively. The four-class model was trained to discriminate between *road-damage*, *building-damage*, *bridge-damage* and *no-damage* images. Finally, the six-class model was trained to discriminate between *building-damage*, *no-building-damage*, *bridge-damage*, *no-bridge-damage*, *road-damage*, and *no-road-damage*.



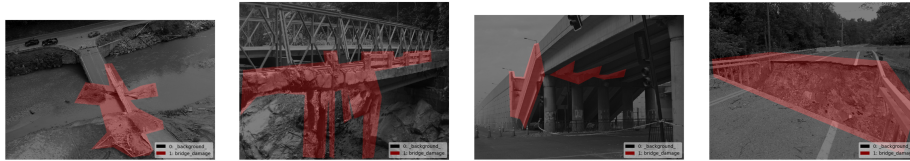
a (a) Original *building-damage* images



b (b) Annotated *building-damage* images



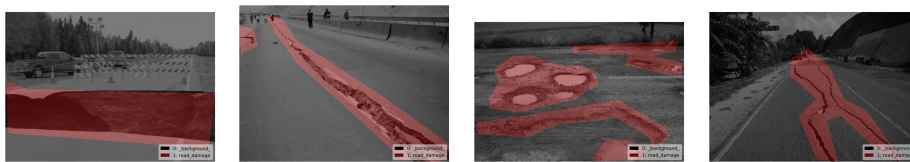
c (c) Original *bridge-damage* images



d (d) Annotated *bridge-damage* images



e (e) Original *road-damage* images



f (f) Annotated *road-damage* images

Figure 5.6: Sample images together with their corresponding damage localization annotation for *building-damage*, *bridge-damage* and *road-damage*, respectively.

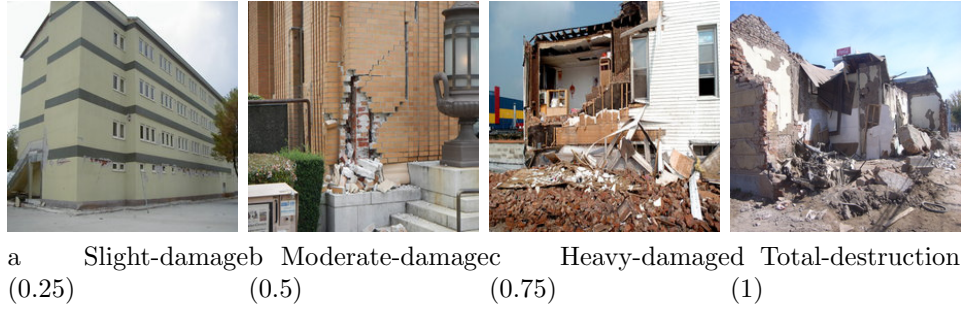


Figure 5.7: Sample images from the Building Damage Severity dataset, together with their severity annotations

Hyper-parameters

We used TensorFlow’s GradientDescentOptimizer to train the model using mini-batch gradient descent on a GeForce GTX 1070 graphic card. Based on preliminary experimentation, we chose to use a learning rate of 0.001 and a batch size of 32 images in all our experiments. Furthermore, we used the dropout technique with a rate of 0.5 to prevent overfitting. The code for the VGG19 model was adapted from <https://github.com/machrisaa/tensorflow-vgg>.

Evaluation Metrics

We used several metrics to evaluate the performance of the proposed approaches on the test data. First, we evaluated the performance of the VGG19 models using the classification accuracy (by comparing the predicted image labels with the ground truth labels). The ability of the Grad-CAM and Grad-CAM++ approaches to produce DDM heatmaps that accurately localize the damage was evaluated using the average intersection-over-union (IoU) metric [110] over all correctly classified images.

To compare heatmaps with images where damage is manually marked in terms of IOU values, we transform the heatmaps to a binary representation as follows: we determine the maximum value in S_C and use 20% of the maximum value as a cutoff value for including a region in the disaster damage “object” or not [42]. In other words, only the regions in DDM with values larger than the cutoff value will be part of the localized damage. In the

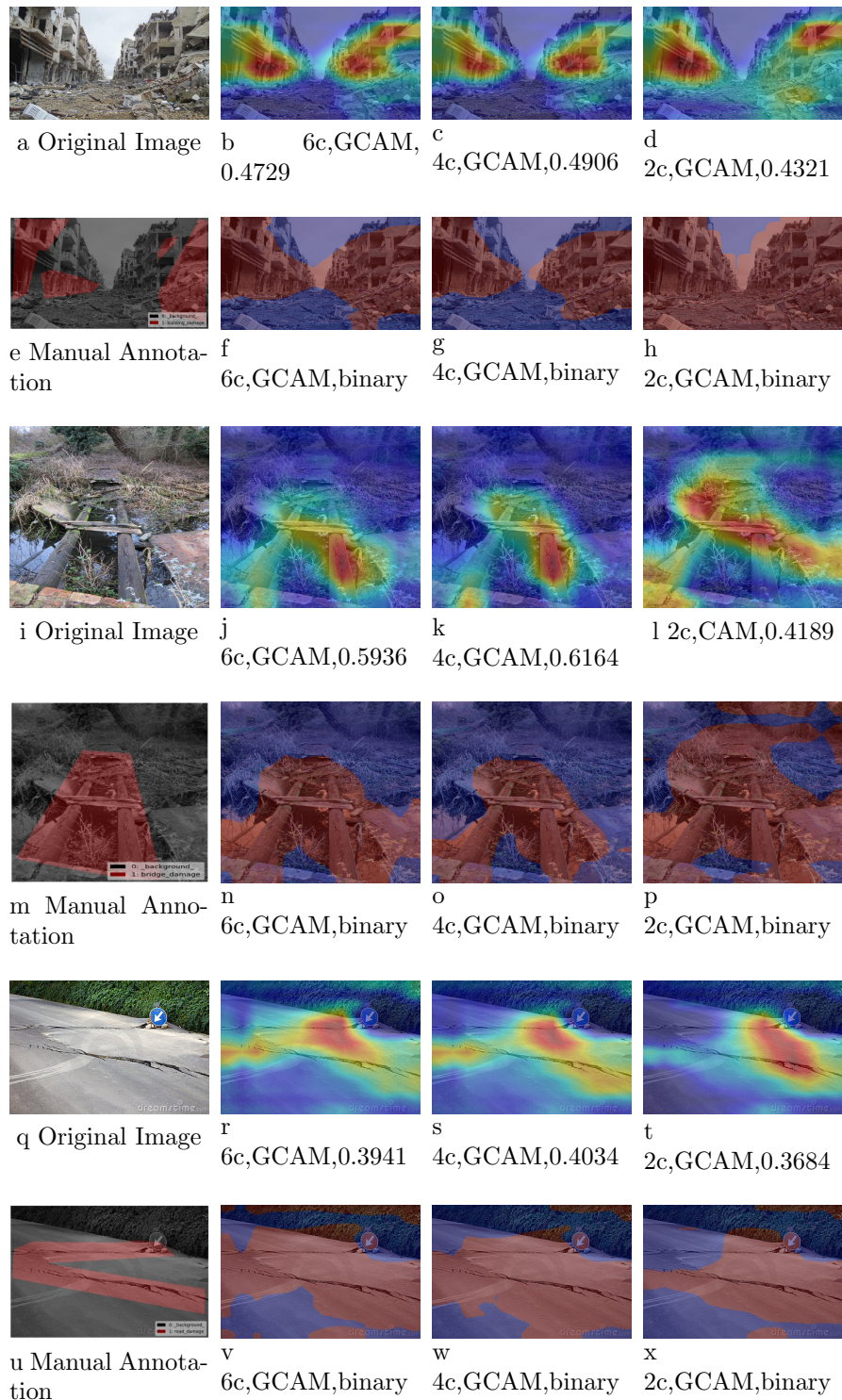


Figure 5.8: Visualization of Damage Detection Maps for a building image (top two rows), a bridge image (middle two rows) and a road image (last two rows). The first row corresponding to an image shows the original image and the DDM maps produced with a six-class (6c), four-class (4c) and two-class (2c) VGG19 model, respectively. The Grad-CAM (GCAM) is used with all models. For each model, the IoU value of the image is also shown. The second row corresponding to an image shows the manual damage annotation of the image, together with the binary maps obtained from the DDM heatmaps (the binary maps are used to calculate the IoU values).

resulting transformed image (as well as in the human annotated images), the damage pixels have value 255, while no damage pixels have value 0.

To evaluate the ability of the proposed approaches to properly detect “damage areas/objects” we consider different IoU detection thresholds k (i.e., an object is detected if its IoU with the ground object is greater than the threshold k), and calculate the average precision over the detected objects. This metric is denoted as $AP@k$.

Finally, the ability of the DAV scores to capture the severity of the damage was evaluated using the root mean square error (RMSE).

5.4.2 Experimental Results and Discussion

In this section, we show the results of the experiments for evaluating the damage detection maps produced by our approach on the bridge, building and road categories, as well as the damage assessment value computed based on the damage detection maps. Finally, we use the results of the experiments to answer the research questions that we raised in Section 5.4.1.

Damage Detection Map Evaluation

The results of the experiments on the Google test dataset are shown in Table 5.6. The table first shows the accuracy of the VGG19 models used on the test data, as intuitively, a more accurate model will lead to a better DDM map. As can be seen in the table, the two-class VGG19 models are more accurate than the four-class and six-class models. The two-class model that discriminates between *road-damage* and *no-road-damage* is the most accurate among the two-class models, while the six-class model has the lowest accuracy overall. The two-class road model is also the best in terms of the average IoU regardless of the CAM approach used to construct the DDM heatmap, but the six-class model is slightly better than the four-class model. The two-class bridge model has worse IoU values as compared to the other models. We should also note that the standard deviation for all IoU values is relatively high, which means that some images have high IoU, while others have

low IoU. A sample image in each category, together with its manual annotation, the DDM heatmaps obtained with different models, and the corresponding binary maps are shown in Figure 5.8. As can be seen, the heatmaps produced by different models are not very different, but they lead to different binary maps and consequently different IoU values. Finally, it can be seen that overall the Grad-CAM approach leads to better average IoU as compared to the Grad-CAM++ approach, regardless of the model used.

In addition to accuracy and average IoU, Table 5.6 also shows the average precision for each category of interest (i.e., bridge, building, and road) at different IoU thresholds k (specifically, $k = 0.5, 0.4,$ and 0.3) with different VGG19 models (six-class, four-class and two-class) and different CAM approaches (Grad-CAM and Grad-CAM++). For a particular VGG19 model, one of the two CAM approaches is used to create a heatmap. Subsequently, a “damage object” is detected by taking a threshold on the values in the heatmap, as explained in Section 5.4.1. The IoU between the detected object and the corresponding ground truth object is computed. If the IoU is greater than the threshold k , the object is considered to be a true positive, otherwise it is considered to be a false positive. This information is used to compute $AP@k$. Intuitively, a higher threshold k may lead to some objects not being detected, while a lower threshold k may lead to false positives. As can be seen in Table 5.6, when using the six-class and four-class models, for the *bridge* category the $AP@0.4$ is worse than then $AP@0.5$ and also worse than $AP@0.3$, while for the other two categories, the values increase as k decreases. A similar pattern is consistently observed for the two-class models, where the performance increases as k decreases. It can also be seen that Grad-CAM generally gives better results than Grad-CAM++. When comparing the models for a particular category, we can see that the two-class road model is competitive in terms of average precision for the road category, although the four-class model perform well on this category as well. However, the six-class model is the best overall in terms of average precision for the bridge and building categories.

Damage Assessment Value Evaluation

To evaluate the ability of the DAV scores to capture damage severity, we used the two-class VGG19 model fine-tuned to discriminate between *building-damage* and *no-building-damage* and tested it on the *Building Damage Severity* dataset. More specifically, we first used the two-class VGG19 building model and Grad-CAM/Grad-CAM++ to create a re-scaled DDM heatmap S_C for each test image in the *Building Damage Severity* dataset. Subsequently, we used the heatmap S_C to compute a DAV score for each test image. Finally, we computed the RMSE over the set of images in the *Building Damage Severity* dataset, by comparing the DAV score of an image with the numeric annotation of the image (which captures damage severity). The resulting RMSE value for the Grad-CAM approach is 0.2538, while the RMSE value for the Grad-CAM++ approach is 0.4189, a result which further confirms the previous finding that Grad-CAM performs better than Grad-CAM++ in terms of damage detection and quantification.

The DAV values for the sample images in Fig. 5.7 are shown below the corresponding heatmap images in Fig. 5.9. As can be seen, images that present a more severe damage scene have higher DAV values, while images with less damage have smaller DAV scores. Thus, the DAV scores accurately reflect the degree of damage.

Discussion of the Results

Given the results presented above, in this subsection we answer the research questions that we raised in Section 5.4.1.

1. *Can the Damage Detection Map accurately locate the damage areas?* Conventionally, if the IoU value corresponding to a detected object (marked with a bounding box) is larger than 0.5, the detection/localization of that object is considered to be correct [110]. However, given that the damage is not an object but a concept, we find that the 0.5 threshold is too strict in the context of detecting and localizing damage. To explain this, we got an estimate for the average IoU agreement between two human annotators. Specifically, we had a second annotator mark damage for a random subset

	six-class	four-class	two-class	two-class	two-class
	all	all	bridge	building	road
Accuracy	0.8359	0.8700	0.8714	0.9198	0.9550
Avg. IoU (Grad-CAM)	0.3299	0.3132	0.2477	0.3361	0.4184
Std. IoU (Grad-CAM)	0.1758	0.1837	0.1575	0.1971	0.1745
Avg. IoU (Grad-CAM++)	0.3148	0.3054	0.2345	0.3366	0.3836
Std. IoU (Grad-CAM++)	0.1825	0.1876	0.1565	0.1926	0.1644
AP@0.5 bridge (Grad-CAM)	0.3167	0.2935	0.2803	N/A	N/A
AP@0.5 building (Grad-CAM)	0.5221	0.3493	N/A	0.4424	N/A
AP@0.5 road (Grad-CAM)	0.2454	0.3049	N/A	N/A	0.2718
AP@0.5 bridge (Grad-CAM++)	0.3167	0.2934	0.2038	N/A	N/A
AP@0.5 building (Grad-CAM++)	0.5156	0.3560	N/A	0.4559	N/A
AP@0.5 road (Grad-CAM++)	0.2416	0.2516	N/A	N/A	0.2725
AP@0.4 bridge (Grad-CAM)	0.5738	0.5526	0.2006	N/A	N/A
AP@0.4 building (Grad-CAM)	0.5425	0.5159	N/A	0.4729	N/A
AP@0.4 road (Grad-CAM)	0.4172	0.4678	N/A	N/A	0.4547
AP@0.4 bridge (Grad-CAM++)	0.2349	0.2934	0.2146	N/A	N/A
AP@0.4 building (Grad-CAM++)	0.5420	0.5160	N/A	0.4943	N/A
AP@0.4 road (Grad-CAM++)	0.3749	0.4757	N/A	N/A	0.3952
AP@0.3 bridge (Grad-CAM)	0.4537	0.4270	0.5005	N/A	N/A
AP@0.3 building (Grad-CAM)	0.7267	0.7098	N/A	0.6049	N/A
AP@0.3 road (Grad-CAM)	0.5881	0.6998	N/A	N/A	0.7222
AP@0.3 bridge (Grad-CAM++)	0.4537	0.4270	0.4455	N/A	N/A
AP@0.3 building (Grad-CAM++)	0.7178	0.6639	N/A	0.6113	N/A
AP@0.3 road (Grad-CAM++)	0.5589	0.6558	N/A	N/A	0.6918

Table 5.6: Experimental results on the Google test dataset

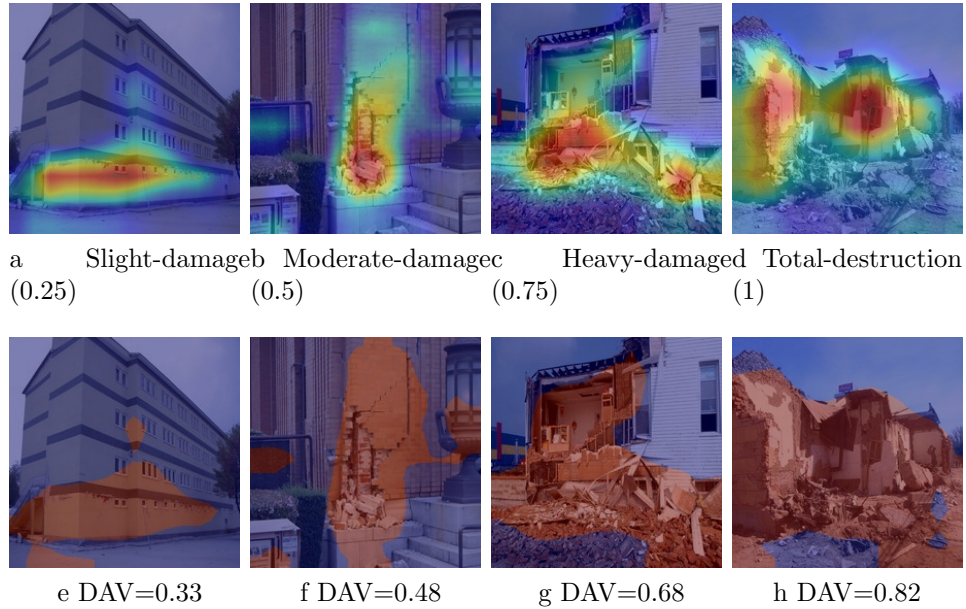


Figure 5.9: Heatmaps (top row) and the corresponding binary maps (bottom row) for the sample images from the Building Damage Severity dataset, together with their severity annotations (shown below the top images) and the DAV scores (shown below the bottom images). The DAV scores properly reflect the severity of the damage.

of the images in our dataset (precisely, 768 images). The average IoU between the two manual annotations was 0.5356, with standard deviation 0.2323 (larger than the standard deviation obtained with the automated approaches). Furthermore, recall that that the average precision for different categories was generally best for IoU threshold $k = 0.3$. This result, together with the relatively low agreement between annotators and the large standard deviation, suggests that the average IoU obtained using the automated approaches, which is as high as 0.4184 for road damage, is an indication that the damage detection map does a reasonable job at identifying infrastructure damage. This can also be seen from the samples images in Figure 5.10, which shows both images with DDMs that produce high IoU values, and images with DDMs that produce low IoU values. As can be seen, for some of the images with low IoU values (e.g., the building image), one can argue that the annotations may not best capture the damage area in the first place, while for others (e.g., the bridge image), the model fails to annotate the most relevant damage area.

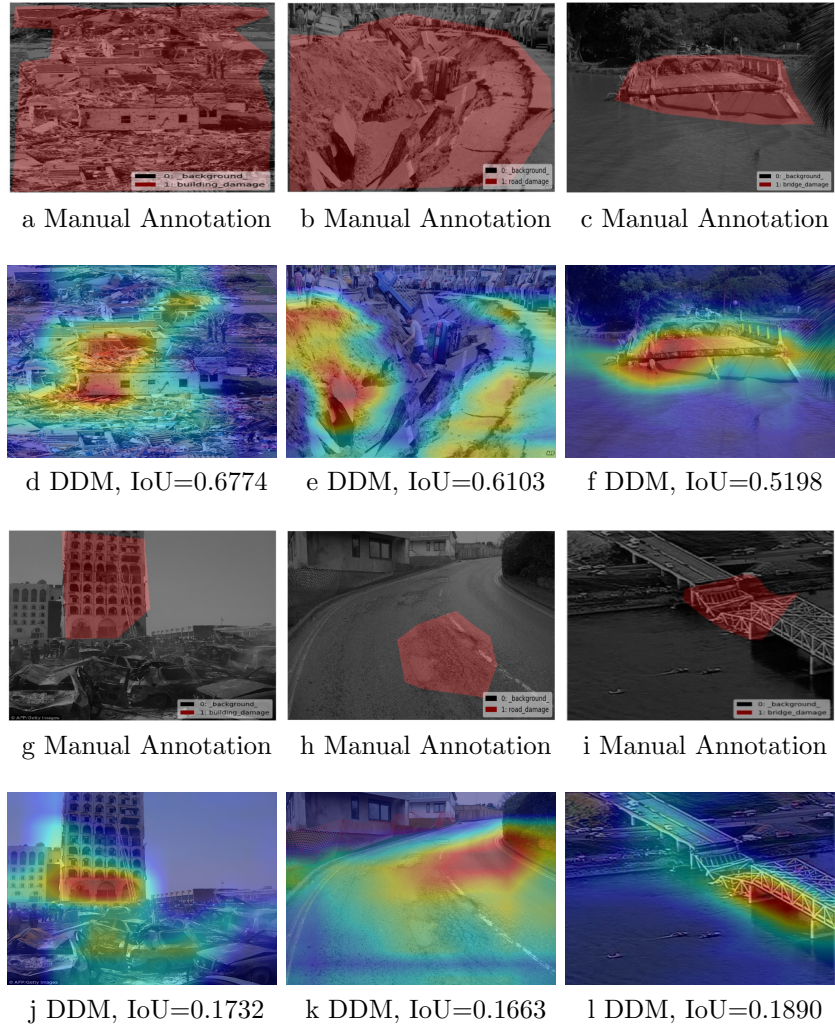


Figure 5.10: Examples of images with DDM heatmaps that lead to high/low IoU values. The first two rows show images with high IoU values, while the last two rows show images with low IoU values.

2. *What type of damage can be localized more accurately?* The two-class VGG19 road model is the most accurate and produces the highest average IoU values. Furthermore, a high average precision value (specifically, 0.722) is obtained for the road category when using the same two-class VGG19 model and an IoU threshold of 0.3. However, the highest average precision value overall is 0.7267, and it is obtained for the building category with the six-class VGG19 model. Thus, we can say that both road damage and building damage can be detected with high accuracy. For the bridge category, the highest average precision value is 0.5738 and it is also obtained with the six-class VGG

model. One reason for bridge damage being harder to localize may come from the fact that bridge damage can be mistaken for road damage or building damage, as has been observed when analyzing the original labels of the images in the Google dataset.

3. *Can the DAV scores provide a reliable measure for damage severity?* Our results show that the RMSE between the DAV scores produced by our approach and the ground truth damage severity annotations is 0.2538, when the Grad-CAM approach is used to generate the DDM heatmaps and subsequent DAV scores. Given that the damage severity is in the range $[0,1]$, we find this result to be very promising, in other words, we can claim that the DAV score provide a good measure for damage severity, although the reliability can be further improved.
4. *Among two-class, four-class, or six-class models, what model leads to better results overall?* Overall, the six-class VGG19 model together with Grad-CAM works best for our set of images, although the four-class model is also competitive, and the two-class VGG19 road model is the best for road damage. Intuitively, the six-class model, which is trained to discriminate between *bridge-damage*, *no-bridge-damage*, *building-damage*, *no-building-damage*, *road-damage*, and *no-road-damage*, learns not only features that are indicative of damage versus no-damage, but also features that are indicative of a specific category among the three target categories (bridge, building, road). As opposed to that, the four-class model, which is train to discriminate between *bridge-damage*, *building-damage*, *road-damage*, and *no-damage* may learn features that identify no-damage in general, as opposed to no-damage in a particular category (in addition to features for discriminating between different categories), and thus leads to worse results than the six-class models. While the two-class road model works the best for road damage, as expected, it seems surprising that the two-class bridge and building models are generally worse than the other models. Given that some bridge images look similar to building images, it may be that the richer features extracted from the combined dataset help get better DDM heatmaps. As opposed to that, road damage might be more distinctive and easier to identify with a two-class model.

5. *Between Grad-CAM and Grad-CAM++, which approach is better?* Our results show that the Grad-CAM approach is consistently better than the Grad-CAM++ approach. This may seem surprising, as Grad-CAM++ is an extension of Grad-CAM, which has been shown to give better results on datasets with multiple occurrences of an object in an image. However, in the context of damage localization, damage is rather as a concept with soft boundaries rather than an object with hard boundaries, and as such, it doesn't present multiple occurrences. The Grad-CAM++ may highlight additional regions that are not representative of damage, as marked by human annotators. Therefore, Grad-CAM++ can lead to worse results as compared to Grad-CAM, which focuses on the most representative regions for the damage.

5.5 Related Work

Social media data has been shown to have significant value in disaster response [112, 113, 114]. Many machine learning approaches [87, 80, 64], including deep learning approaches [54, 4], have been proposed and used to help identify and prioritize useful *textual* information (e.g., tweets) in social media. Some works have focused specifically on identifying situational awareness information [51, 52], including information related to damage assessment [48, 101, 115, 116].

Despite the extensive use of machine learning tools for analyzing social media text data posted during disaster events, there is not much work on analyzing social media images posted by eyewitnesses of a disaster. One pioneering work in this area [18], used trained CNN models, specifically, VGG16 networks fine-tuned on the disaster image datasets that are also used in our work (see Table 5.1), and showed that the CNN models perform better than standard techniques based on bags-of-visual-words. Our CNN results, using VGG19 fine-tuned on the same disaster image datasets, are similar to those reported in [18], except for Matthew Hurricane, for which the dataset is relatively small and the model can't be trained well. However, as opposed to [18], where the focus is on classifying images into three damage categories, we go one step further and use the Grad-CAM approach [106] to localize

damage, with the goal of improving the trust of the response teams in the predictions of the model. Furthermore, we produce a continuous severity score, as a way to quantify damage.

Other prior works focused on image-based disaster damage assessment use aerial or satellite images, e.g. [96, 97]. Compared to such works, which use more expensive imagery, we focus on the use of social media images, which are readily available during disasters, together with interpretability approaches, i.e., Grad-CAM, to produce a damage map and a damage severity score for each image.

Similar to us, Nia and Mori [111] use ground-level images collected using Google to assess building damage. Their model consists of three different CNN networks (fine-tuned with raw images, color-masked or binary-masked images, respectively) to extract features predictive of damage. Subsequently, a regression model is used with the extracted features to predict the severity of the damage on a continuous scale. Compared to [111], we use the features identified at the last convolutional layer of the CNN network to build a detection map, and use the map to produce a numeric damage severity score. While CAM-type approaches have been used to explain model predictions in many other application domains, to the best of our knowledge, they have not been used to locate damage and assess damage severity in prior work.

5.6 Conclusion

Given the large number of social media images posted by eyewitnesses of disasters, we proposed an approach for detecting and localizing disaster damage at low cost. Our approach is built on top of a fine-tuned VGG19 model, and utilizes the Grad-CAM approach to produce a DDM heatmap. Furthermore, the DDM is used to calculate a DAV score for each image. This scoring is performed on a continuous scale and can be used to assess the severity of the damage. The DAV score, together with the DDM heatmap, can be used to identify and prioritize useful information for disaster response, while providing visual explanations for the suggestions made to increase the trust in the computational models. Quantitative and qualitative evaluations of DDM and DAV components show the feasibility of our proposed

approach.

As part of future work, it is of interest to study the applicability of the proposed approach to estimate the global damage produced by a disaster based on aggregating the DAV values from individual images. Also, geo-tagging images would enable disaster response teams not only to identify damage, but also to find its physical location.

Chapter 6

Conclusions and Future Work

With the fast growth of deep learning methods and their successful application in many domains, including image processing and natural language processing [9], it is of great interest to use deep learning to analyze social media data posted by users during disaster events. In this dissertation, we propose several deep learning based approaches to analyze disaster data posted on the social media platform Twitter.

Firstly, we propose two domain adaptation approaches to identify informative text and images, respectively. To achieve near real-time analysis, we propose a domain adaptation based approach that takes advantage of pre-labeled disaster data from a source disaster to filter data for an ongoing target disaster for which labeled data is not available. For text data analysis, we adapt the Domain Reconstruction approach, which learns tweet representations from the target disaster data using a reconstruction approach. Each word in source data and target data is represented by a word embedding which captures dependencies between words. The embeddings and the parameters in different layers together capture the intrinsic information in a tweet. The classifier trained on source data, which uses the representation learned from the target data, can be applied to label the target data. For image analysis, we adapt a Domain Adversarial approach which extracts common features between source data and target data. This approach contains a discriminative component and a generative component. The discriminative component identifies the source of the data (source disaster or

target disaster); the generative component reduces the difference between the representation of the source data and target data so that the discriminative component cannot separate it (thus, the name adversarial). After this adversarial process achieves a balance, the representations of source data and target data become similar, and the classifier trained on the source representation can be applied on the target representation. This approach is evaluated on the images collected during four disasters. Experimental results of the two domain adaptation approaches show that the proposed methods can be used to identify informative text and images efficiently and effectively. Furthermore, since text and images are usually posted together, we propose a multimodal approach that combines the information from text and images. Our approach combines a CNN model for images and an RNN model for text. The last layers of the two models are concatenated and another fully connected layer is applied on the concatenated layer. We study several scenarios to address the challenge that the text and image may have different labels. Experimental results show that the text classification performance can be improved by adding the image information and the image classification performance can be improved by adding the text information.

After the informative tweets and images have been filtered, we propose a damage localization approach to identify the type of damage and the area of damage in images. We introduce the Grad-CAM approach in disaster image analysis to generate a heatmap that highlights the damaged area. This approach trains a VGG model to classify images into damage/no-damage first. The trained model contains key features that classify the images into different categories. In our case, the key features show the damage/no-damage area. The Grad-CAM approach helps to extract these key features from the VGG model. Since the model is trained by gradient descent, the idea of Grad-CAM is to find the area which has the largest gradient. The area with the highest gradient will give the key features that show damage. Then the heatmap can be calculated with the values of the gradients. Furthermore, the value which assesses the severity of damage in images is calculated on top of the heatmap. This approach was evaluated on the images collected during four disasters. As an extension of this work, we evaluated the performance of Grad-CAM++ which is an extension of Grad-CAM. Both methods are evaluated also on a larger dataset. Specifically,

we label a dataset downloaded using google images search engine. The dataset contains several types of infrastructure damage, including building damage, bridge damage, and road damage. Our proposed approaches show promising results on the damage localization and damage evaluation tasks. Experimental results also show promising results for localizing and evaluating the different types of damages.

We propose several future extension directions for the approaches in this dissertation. For the domain adaptation approach for text classification, we plan to evaluate other approaches for the reconstruction framework. For example, we plan to use the Transformer model, which is another autoencoder model similar to the Seq2seq model. For the domain adaptation approach on image classification, we plan to explore the addition of the domain reconstruction method in addition to the current domain adversarial framework. These changes may lead to better performance by enabling the model to learn more information from the target data. For the multimodal approach, we plan to extend it to a domain adaptation approach framework. This will allow us to conduct a near-real-time analysis on the combination of text and image data for the ongoing disaster. Finally, we also plan to evaluate the proposed models on other classification tasks, such as identifying eyewitness tweets and classifying tweets into situational awareness categories, in addition to filtering the informative text and images. With more experiments on other tasks, the proposed models can be used to build an end-to-end solution for disaster data analysis.

Bibliography

- [1] L. Van Der Maaten, “Accelerating t-sne using tree-based algorithms.” *Journal of machine learning research*, vol. 15, no. 1, pp. 3221–3245, 2014.
- [2] F. Alam, S. Joty, and M. Imran, “Domain adaptation with adversarial training and graph embeddings,” in *56th Annual Meeting of the Association for Computational Linguistics*, ser. ACL 2018, 2018.
- [3] F. Alam, F. Offi, and M. Imran, “Crisismmd: Multimodal twitter datasets from natural disasters,” in *Proc. of the International AAAI Conference on Web and Social Media*, ser. ICWSM, Stanford, California, USA, 2018.
- [4] D. T. Nguyen, K. Al-Mannai, S. R. Joty, H. Sajjad, M. Imran, and P. Mitra, “Rapid classification of crisis-related data on social networks using convolutional neural networks,” *CoRR*, vol. abs/1608.03902, 2016.
- [5] M. E. Keim and E. Noji, “Emergent use of social media: a new age of opportunity for disaster resilience.” *American journal of disaster medicine*, vol. 6, no. 1, pp. 47–54, 2011.
- [6] A. Kongthon, C. Haruechaiyasak, J. Pailai, and S. Kongyoung, “The role of social media during a natural disaster: A case study of the 2011 thai flood,” *International Journal of Innovation and Technology Management*, vol. 11, no. 03, p. 1440012, 2014.
- [7] D. Velez and P. Zlateva, “Use of social media in natural disaster management,” *Intl. Proc. of Economic Development and Research*, vol. 39, pp. 41–45, 2012.
- [8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

- [9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [10] H. Li, D. Caragea, C. Caragea, and N. Herndon, “Disaster response aided by tweet classification with a domain adaptation approach,” *Journal of Contingencies and Crisis Management*, vol. 26, no. 1, pp. 16–27, 2018.
- [11] R. Mazloom, H. Li, D. Caragea, C. Caragea, and M. Imran, “A hybrid domain adaptation approach for identifying crisis-relevant tweets,” *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)*, vol. 11, no. 2, pp. 1–19, 2019.
- [12] H. Li, O. Sopova, D. Caragea, and C. Caragea, “Domain adaptation for crisis data using correlation alignment and self-training,” *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)*, vol. 10, no. 4, pp. 1–20, 2018.
- [13] H. Li, X. Li, D. Caragea, and C. Caragea, “Comparison of word embeddings and sentence encodings as generalized representations for crisis tweet classification tasks,” 2018.
- [14] I. Stefan, T. Rebedea, and D. Caragea, “Classification of eyewitness tweets in emergency situations.” in *RoCHI*, 2019, pp. 46–52.
- [15] K. Zahra, M. Imran, and F. O. Ostermann, “Automatic identification of eyewitness messages on twitter during disasters,” *Information processing & management*, vol. 57, no. 1, p. 102107, 2020.
- [16] A. Karami, V. Shah, R. Vaezi, and A. Bansal, “Twitter speaks: A case of national disaster situational awareness,” *Journal of Information Science*, vol. 46, no. 3, pp. 313–324, 2020.
- [17] S. Verma, S. Vieweg, W. J. Corvey, L. Palen, J. H. Martin, M. Palmer, A. Schram, and K. M. Anderson, “Natural language processing to the rescue? extracting” situational

- awareness” tweets during mass emergency,” in *Fifth international AAAI conference on weblogs and social media*, 2011.
- [18] D. T. Nguyen, F. Ofli, M. Imran, and P. Mitra, “Damage assessment from social media imagery data during disasters,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 2017, pp. 569–576.
- [19] Y. LeCun, Y. Bengio *et al.*, “Convolutional networks for images, speech, and time series,” *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.
- [20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [21] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [23] D. E. Rumelhart, G. E. Hinton, R. J. Williams *et al.*, “Learning representations by back-propagating errors,” *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.
- [24] P. Liu, X. Qiu, and X. Huang, “Recurrent neural network for text classification with multi-task learning,” *arXiv preprint arXiv:1605.05101*, 2016.
- [25] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and

- Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [27] V. K. Neppalli, C. Caragea, and D. Caragea, “Deep neural networks versus naive bayes classifiers for identifying informative tweets during disasters,” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, Rochester, NY, 2018.
- [28] A. Hu and S. Flaxman, “Multimodal sentiment analysis to explore the structure of emotions,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 350–358.
- [29] H. Mouzannar, Y. Rizk, and M. Awad, “Damage identification in social media posts using multimodal deep learning.” in *ISCRAM*, 2018.
- [30] G. Nalluru, R. Pandey, and H. Purohit, “Relevancy classification of multimodal social media streams for emergency services,” in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2019, pp. 121–125.
- [31] S. Sun, H. Shi, and Y. Wu, “A survey of multi-source domain adaptation,” *Information Fusion*, vol. 24, pp. 84–92, 2015.
- [32] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *JMLR*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [33] X. Fang, H. Bai, Z. Guo, B. Shen, S. Hoi, and Z. Xu, “Dart: Domain-adversarial residual-transfer networks for unsupervised cross-domain image classification,” *arXiv preprint arXiv:1812.11478*, 2018.
- [34] R. Gong, W. Li, Y. Chen, and L. Van Gool, “Dlow: Domain flow for adaptation and generalization,” *arXiv preprint arXiv:1812.05418*, 2018.

- [35] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, “Cycada: Cycle-consistent adversarial domain adaptation,” *arXiv preprint arXiv:1711.03213*, 2017.
- [36] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, “Deep reconstruction-classification networks for unsupervised domain adaptation,” in *European Conference on Computer Vision*. Springer, 2016, pp. 597–613.
- [37] X. Li, D. Caragea, C. Caragea, M. Imran, and F. Ofli, “Identifying disaster damage images using a domain adaptation approach,” in *Proceedings of the 16th International Conference on Information Systems for Crisis Response and Management (ISCRAM), Valencia, Spain*. Academic Press, 2019.
- [38] X. Li and D. Caragea, “Domain adaptation with reconstruction for disaster tweet classification,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1561–1564.
- [39] —, “Improving disaster-related tweets classification with multimodal,” in *Proceedings of the 17th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, 2020.
- [40] X. Li, D. Caragea, H. Zhang, and M. Imran, “Localizing and quantifying damage in social media images,” in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 194–201.
- [41] —, “Localizing and quantifying infrastructure damage using class activation mapping approaches,” *Social Network Analysis and Mining*, vol. 9, no. 1, p. 44, 2019.
- [42] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2921–2929.

- [43] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, “Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks,” *CoRR*, vol. abs/1710.11063, 2017.
- [44] M. Rhodan, “‘please send help.’ hurricane harvey victims turn to twitter and facebook,” *Time*, 2017-08-30, Available at <http://time.com/4921961/hurricane-harvey-twitter-facebook-social-media/>, 2017.
- [45] D. MacMillan, “In irma, emergency responders new tools: Twitter and facebook,” *The Wall Street Journal*, 2017-09-11, Available at <https://www.wsj.com/articles/for-hurricane-irma-information-officials-post-on-social-media-1505149661>, 2017.
- [46] W. Frej, “Hurricane florence flood victims turn to social media for rescue,” *HuffPost*, 2018-09-14, Available at https://www.huffingtonpost.com/entry/hurricane-florence-flood-victims-social-media_us_5b9b86c0e4b013b097799cd1, 2018.
- [47] C. Villegas, M. Martinez, and M. Krause, “Lessons from harvey: Crisis informatics for urban resilience,” *Rice University Kinder Institute for Urban Research*, 2018.
- [48] Y. Kryvasheyev, H. Chen, N. Obradovich, E. Moro, P. Van Hentenryck, J. Fowler, and M. Cebrian, “Rapid assessment of disaster damage using social media activity,” *Science advances*, vol. 2, no. 3, 2016.
- [49] M. Enenkel, S. M. Saenz, D. S. Dookie, L. Braman, N. Obradovich, and Y. Kryvasheyev, “Social media data analysis and feedback for advanced disaster risk management,” in *Social Web in Emergency and Disaster Management*, 2018.
- [50] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen, “Microblogging during two natural hazards events: what twitter may contribute to situational awareness,” in *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 2010, pp. 1079–1088.
- [51] A. Sen, K. Rudra, and S. Ghosh, “Extracting situational awareness from microblogs

- during disaster events,” in *Communication Systems and Networks (COMSNETS), 2015 7th International Conference on*. IEEE, 2015, pp. 1–6.
- [52] Q. Huang and Y. Xiao, “Geographic situational awareness: mining tweets for disaster preparedness, emergency response, impact, and recovery,” *ISPRS Int. Journal of Geo-Information*, vol. 4, no. 3, pp. 1549–1568, 2015.
- [53] M. Imran, S. Chawla, and C. Castillo, “A robust framework for classifying evolving document streams in an expert-machine-crowd setting,” in *Proc. of the 18th Int. Conf. on Data Mining (ICDM)*, Barcelona, Spain, 2016.
- [54] C. Caragea, A. Silvescu, and A. H. Tapia, “Identifying informative messages in disasters using convolutional neural networks,” in *13th Proceedings of the International Conference on Information Systems for Crisis Response and Management, Rio de Janeiro, Brasil, May 22-25, 2016.*, 2016.
- [55] D. T. Nguyen, K. Al-Mannai, S. R. Joty, H. Sajjad, M. Imran, and P. Mitra, “Robust classification of crisis-related data on social networks using convolutional neural networks,” in *11th International AAAI Conference on Web and Social Media*, ser. ICWSM 2017, Montreal, CA, 2017.
- [56] L. Derczynski, K. Meesters, K. Bontcheva, and D. Maynard, “Helping crisis responders find the informative needle in the tweet haystack,” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, Rochester, NY, 2018.
- [57] A. Aipe, N. Mukuntha, A. Ekbal, and S. Kurohashi, “Deep learning approach towards multi-label classification of crisis related tweets,” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management*, ser. ISCRAM 2018, Rochester, NY, 2018.
- [58] M. Bica, L. Palen, and C. Bopp, “Visual representations of disaster,” in *Proceedings*

of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, ser. CSCW '17, 2017, pp. 1262–1276.

- [59] R. Lagerstrom, Y. Arzhaeva, P. Szul, O. Obst, R. Power, B. Robinson, and T. Bednarz, “Image classification to support emergency situation awareness,” *Frontiers in Robotics and AI*, vol. 3, p. 54, 2016.
- [60] F. Alam, F. Ofli, and M. Imran, “Processing social media images by combining human and machine computing during crises,” *International Journal of Human-Computer Interaction*, vol. 34, no. 4, pp. 311–327, 2018.
- [61] H. Mouzannar, Y. Rizk, and M. Awad, “Damage identification in social media posts using multimodal deep learning,” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, Rochester, NY, 2018.
- [62] D. T. Nguyen, F. Ofli, M. Imran, and P. Mitra, “Damage assessment from social media imagery data during disasters,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ser. ASONAM '17. New York, NY, USA: ACM, 2017, pp. 569–576.
- [63] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [64] H. Li, D. Caragea, C. Caragea, and N. Herndon, “Disaster response aided by tweet classification with a domain adaptation approach,” *Journal of Contingencies and Crisis Management (JCCM), Special Issue on HCI in Critical Systems.*, vol. 26, no. 1, pp. 16–27, 2017.
- [65] R. Mazloom, H. Li, D. Caragea, M. Imran, and C. Caragea, “Classification of twitter disaster data using a hybrid feature-instance adaptation approach,” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, Rochester, NY, 2018.

- [66] M. Wang and W. Deng, “Deep visual domain adaptation: A survey,” *Neurocomputing*, vol. 312, pp. 135 – 153, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0925231218306684>
- [67] R. Gopalan, R. Li, and R. Chellappa, “Domain adaptation for object recognition: An unsupervised approach,” in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 999–1006.
- [68] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a ”siamese” time delay neural network,” in *Proceedings of the 6th International Conference on Neural Information Processing Systems*, ser. NIPS’93. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993, pp. 737–744. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2987189.2987282>
- [69] M. Long, Y. Cao, J. Wang, and M. I. Jordan, “Learning transferable features with deep adaptation networks,” in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML’15. JMLR.org, 2015, pp. 97–105. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3045118.3045130>
- [70] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [71] D. Yarowsky, “Unsupervised word sense disambiguation rivaling supervised methods,” in *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ser. ACL ’95. Stroudsburg, PA, USA: Association for Computational Linguistics, 1995, pp. 189–196.
- [72] D. Bang and H. Shim, “Improved training of generative adversarial networks using representative features,” in *Proceedings of the 35th International Conference on Machine*

- Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmssan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 433–442.
- [73] D. T. Nguyen, F. Alam, F. Ofi, and M. Imran, “Automatic image filtering on social networks using deep learning and perceptual hashing during crises,” in *Proceedings of the 14th International Conference on Information Systems for Crisis Response and Management*, ser. ISCRAM 2017, Albi, France, 2017.
- [74] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, “Analysis of representations for domain adaptation,” in *Advances in neural information processing systems*, 2007, pp. 137–144.
- [75] J. B. Houston, J. Hawthorne, M. F. Perreault, E. H. Park, M. Goldstein Hode, M. R. Halliwell, S. E. Turner McGowen, R. Davis, S. Vaid, J. A. McElderry *et al.*, “Social media and disasters: a functional framework for social media use in disaster planning, response, and research,” *Disasters*, vol. 39, no. 1, pp. 1–22, 2015.
- [76] N. Dufty *et al.*, “Using social media to build community disaster resilience,” *Australian Journal of Emergency Management, The*, vol. 27, no. 1, p. 40, 2012.
- [77] C. Reuter and T. Spielhofer, “Towards social resilience: A quantitative and qualitative survey on citizens’ perception of social media in emergencies in europe,” *Technological Forecasting and Social Change*, vol. 121, pp. 168–180, 2017.
- [78] J. Rogstadius, M. Vukovic, C. A. Teixeira, V. Kostakos, E. Karapanos, and J. A. Laredo, “Crisistracker: Crowdsourced social media curation for disaster awareness,” *IBM Journal of Research and Development*, vol. 57, no. 5, pp. 4–1, 2013.
- [79] D. E. Alexander, “Social media in disaster risk reduction and crisis management,” *Science and engineering ethics*, vol. 20, no. 3, pp. 717–733, 2014.
- [80] M. Imran, C. Castillo, F. Diaz, and S. Vieweg, “Processing social media messages in mass emergency: A survey,” *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, p. 67, 2015.

- [81] D. T. Nguyen, K. A. Al Mannai, S. Joty, H. Sajjad, M. Imran, and P. Mitra, “Robust classification of crisis-related data on social networks using convolutional neural networks,” in *ICWSM*, 2017.
- [82] C. Reuter, A. L. Hughes, and M.-A. Kaufhold, “Social media in crisis management: An evaluation and analysis of crisis informatics research,” *Int. J. of HumanComputer Interaction*, vol. 34, no. 4, pp. 280–294, 2018.
- [83] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.
- [84] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in neural information processing systems*, 2014, pp. 3104–3112.
- [85] Y. Xiao, Q. Huang, and K. Wu, “Understanding social media data for disaster management,” *Natural hazards*, vol. 79, no. 3, pp. 1663–1679, 2015.
- [86] F. Alam, M. Imran, and F. Offi, “Image4act: Online social media image processing for disaster response,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 2017, pp. 601–604.
- [87] Z. Ashktorab, C. Brown, M. Nandi, and A. Culotta, “Tweedr: Mining twitter to inform disaster response.” in *ISCRAM*, 2014.
- [88] V. K. Neppalli, C. Caragea, and D. Caragea, “Deep neural networks versus naive bayes classifiers for identifying informative tweets during disasters.” in *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, 2018.
- [89] M. Agarwal, M. Leekha, R. Sawhney, and R. R. Shah, “Crisis-dias: Towards multi-modal damage analysis - deployment, challenges and assessment,” 2020, p. Proceedings of the 34th American Association for Artificial Intelligence (AAAI 2020).

- [90] S. R. Chowdhury, M. Imran, M. R. Asghar, S. Amer-Yahia, and C. Castillo, “Tweet4act: Using incident-specific profiles for classifying crisis-related messages.” in *ISCRAM*, 2013.
- [91] K. Stowe, M. Paul, M. Palmer, L. Palen, and K. M. Anderson, “Identifying and categorizing disaster-related tweets,” in *Proceedings of The fourth international workshop on natural language processing for social media*, 2016, pp. 1–6.
- [92] Y. Yang, H.-Y. Ha, F. Fleites, S.-C. Chen, and S. Luis, “Hierarchical disaster image classification for situation report enhancement,” in *2011 IEEE International Conference on Information Reuse & Integration*. IEEE, 2011, pp. 181–186.
- [93] C. F. Barnes, H. Fritz, and J. Yoo, “Hurricane disaster assessments with image-driven data mining in high-resolution satellite imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 6, pp. 1631–1640, 2007.
- [94] A. Vetrivel, N. Kerle, M. Gerke, F. Nex, and G. Vosselman, “Towards automated satellite image segmentation and classification for assessing disaster damage using data-specific features with incremental learning,” 2016.
- [95] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” in *Advances in neural information processing systems*, 2017, pp. 3146–3154.
- [96] S. Xie, J. Duan, S. Liu, Q. Dai, W. Liu, Y. Ma, R. Guo, and C. Ma, “Crowdsourcing rapid assessment of collapsed buildings early after the earthquake based on aerial remote sensing image: A case study of yushu earthquake,” *Remote Sensing*, vol. 8, no. 9, p. 759, 2016.
- [97] L. Gueguen and R. Hamid, “Large-scale damage detection using satellite imagery,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1321–1328.

- [98] Y. Fan, Q. Wen, W. Wang, P. Wang, L. Li, and P. Zhang, “Quantifying disaster physical damage using remote sensing data: a technical work flow and case study of the 2014 Ludian earthquake in China,” *International Journal of Disaster Risk Science*, vol. 8, no. 4, pp. 471–488, 2017.
- [99] N. Attari, F. Offi, M. Awad, J. Lucas, and S. Chawla, “Nazr-cnn: Fine-grained classification of UAV imagery for damage assessment,” in *Data Science and Advanced Analytics (DSAA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 50–59.
- [100] X. Guan and C. Chen, “Using social media data to understand and assess disasters,” *Natural Hazards*, vol. 74, no. 2, pp. 837–850, 2014.
- [101] F. Yuan and R. Liu, “Feasibility study of using crowdsourcing to identify critical affected areas for rapid damage assessment: Hurricane Matthew case study,” *International Journal of Disaster Risk Reduction*, 2018.
- [102] Y.-J. Cha, W. Choi, and O. Büyüköztürk, “Deep learning-based crack damage detection using convolutional neural networks,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017.
- [103] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiwayama, and H. Omata, “Road damage detection using deep neural networks with images captured through a smartphone,” *arXiv preprint arXiv:1801.09454*, 2018.
- [104] A. Vetrivel, M. Gerke, N. Kerle, F. Nex, and G. Vosselman, “Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 140, pp. 45–59, 2018.
- [105] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, “SSD: single shot multibox detector,” in *In European conference on computer vision*. Springer, 2016, p. 2137.

- [106] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, “Grad-cam: Why did you say that?” *arXiv preprint arXiv:1611.07450*, 2016.
- [107] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, 1989.
- [108] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [109] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” in *Advances in neural information processing systems*, 2014, pp. 3320–3328.
- [110] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (voc) challenge,” *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.
- [111] K. R. Nia and G. Mori, “Building damage assessment using deep learning and ground-level image data,” in *14th Conference on Computer and Robot Vision (CRV)*. IEEE, 2017, pp. 95–102.
- [112] C. Castillo, *Big Crisis Data: Social Media in Disasters and Time-Critical Situations*. Cambridge University Press, 2016.
- [113] P. Meier, *Digital Humanitarians: How Big Data Is Changing the Face of Humanitarian Response*. FL, USA: CRC Press, Inc., 2015.
- [114] L. Palen and K. M. Anderson, “Crisis informatics-new data for extraordinary times,” *Science*, vol. 353, no. 6296, pp. 224–225, 2016.
- [115] B. Resch, F. Usländer, and C. Havas, “Combining machine-learning topic models and

- spatiotemporal analysis of social media data for disaster footprint and damage assessment,” *Cartography and Geographic Information Science*, pp. 1–15, 2017.
- [116] M. Enenkel, S. M. Saenz, D. S. Dookie, L. Braman, N. Obradovich, and Y. Kryvasheyev, “Social media data analysis and feedback for advanced disaster risk management,” *CoRR*, vol. abs/1802.02631, 2018.
- [117] J. Lin, “Divergence measures based on the shannon entropy,” *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145–151, 1991.

Appendix A

Optimally of the Adversarial Training

In this supplementary material document, we present theoretical results that show the optimality of the adversarial training component of our domain adaptation model for identifying disaster damage images.

A.1 Global Optimally of $P_{T(X_s)} = P_{T(X_t)}$

Our problem can be seen as a multi-task learning problem. Thus, the loss function \mathcal{L}_{total} can be re-written as:

$$\mathcal{L}_{total} = \mathcal{L}_{adversarial} + \mathcal{L}_{prediction} \tag{A.1}$$

where $\mathcal{L}_{adversarial} = \mathcal{L}_{domain} + \mathcal{L}_{transfer}$ is the loss which needs to be minimized to reduce the difference between the source and target domains, and $\mathcal{L}_{prediction}$ is the loss that needs to be minimized to train the source classifier. We will prove that the adversarial training will lead to source and target representations with similar distributions, i.e., $P_{T(X_s)} = P_{T(X_t)}$. Minimizing the adversarial loss, $\mathcal{L}_{adversarial}$, is equivalent with:

$$\begin{aligned} \min_T \max_D L(T, D) &= E_{x_s} \log(D(T(X_s))) \\ &+ E_{x_t} \log(1 - D(T(X_t))) \end{aligned} \tag{A.2}$$

Here, $D(T(x)) = 0$ if x is from source, and $D(T(x)) = 1$ if x is from target. In practice, we use the one-hot vector representation of different domains. Our optimally proof is inspired from a similar proof by Goodfellow *et al.*[70], who introduced GAN and showed that the generated samples have similar distribution with the real samples.

Theorem 1. *For $T(X)$ fixed, the optimal $D(X)$ is:*

$$D^*(X) = \frac{P_{T(X_s)}}{P_{T(X_s)} + P_{T(X_t)}} \quad (\text{A.3})$$

Proof. Given $T(X)$, the optimal $D(X)$ maximizes a function L , defined as $L(T(X), D(X)) =$

$$\begin{aligned} &= \int P_{X_s} \log(D(T(x_s))) dx_s + \int P_{X_t} \log(1 - D(T(x_t))) dx_t \\ &= \int P_{T(X_s)} \log(D(x)) dx + \int P_{T(X_t)} \log(1 - D(x)) dx \\ &= \int (P_{T(X_s)} \log(D(x)) + P_{T(X_t)} \log(1 - D(x))) dx \end{aligned}$$

The function $a \log(y) + b \log(1 - y)$ achieves its maximum when $y = \frac{a}{a+b}$. Therefore, $D^*(X)$ is given by Equation (A.3). \square

Theorem 2. *The global optimum of $L(T(X), D(X))$ is achieved if and only if $P_{T(X_s)} = P_{T(X_t)}$.*

Proof. By definition: $L(T(X), D(X)) =$

$$\begin{aligned} &= \int P_{X_s} \log(D(T(x_s))) dx_s + \int P_{X_t} \log(1 - D(T(x_t))) dx_t \\ &= \int (-\log 2 + \log 2 + P_{X_s} \log(D(T(x_s)))) dx_s \\ &+ \int (-\log 2 + \log 2 + P_{X_t} \log(D(T(x_t)))) dx_t \\ &= -\log 4 + \int [P_{T(X_s)} \log(2 \cdot D(x)) \\ &+ P_{T(X_t)} \log(2 \cdot (1 - D(x)))] dx \end{aligned}$$

According to Theorem 1, for any given $T(X)$, the optimal $D(X)$ is given by Eq. (A.3).

Therefore, $\max_D L(T(X), D(X)) =$

$$\begin{aligned}
&= -\log 4 + \int \left[P_{T(X_s)} \log \left(\frac{2P_{T(X_s)}}{P_{T(X_s)} + P_{T(X_t)}} \right) \right. \\
&\quad \left. + P_{T(X_t)} \log \left(\frac{2P_{T(X_t)}}{P_{T(X_s)} + P_{T(X_t)}} \right) \right] dx
\end{aligned} \tag{A.4}$$

Lin [117] defined the following divergence between two distribution p_1 and p_2 :

$$K(p_1, p_2) = \sum_i p_i(x) \log \frac{2p_i(x)}{p_1(x) + p_2(x)} \tag{A.5}$$

According to the Shannon inequality, $K(p_1, p_2) \geq 0$, and $K(p_1, p_2) = 0$ if and only if $p_1 = p_2$ [117]. Using this divergence in Equation (9), we have: $\max_D L(T(X), D(X)) = -\log 4 + K(P_{T(X_s)}, P_{T(X_t)}) \leq -\log 4$. Thus, the minimum after $T(X)$ is obtained when $P_{T(X_s)} = P_{T(X_t)}$, which means that the global optimum of $L(T(X), D(X))$ is achieved if and only if $P_{T(X_s)} = P_{T(X_t)}$. \square