Deep Learning from Smart City Data

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Qi Chen

November 2021

Abstract

Rapid urbanisation brings severe challenges on sustainable development and living quality of urban residents. Smart cities develop holistic solutions in the field of urban ecosystems using collected data from different types of Internet of Things (IoT) sources. Today, smart city research and applications have significantly surged as consequences of IoT and machine learning technological enhancement. As advanced machine learning methods, deep learning techniques provide an effective framework which facilitates data mining and knowledge discovery tasks especially in the area of computer vision and natural language processing. In recent years, researchers from various research fields attempted to apply deep learning technologies into smart city applications in order to establish a new smart city era. Much of the research effort on smart city has been made, for example, intelligence transportation, smart healthcare, public safety, etc. Meanwhile, we still face a lot of challenges as the deep learning techniques are still premature for smart city.

In this thesis, we first provide a review of the latest research on the convergence of deep learning and smart city for data processing. The review is conducted from two perspectives: while the technique-oriented view presents the popular and extended deep learning models, the application-oriented view focuses on the representative application domains in smart cities. We then focus on two areas, which are intelligence transportation and social media analysis, to demonstrate how deep learning could be used in real-world applications by addressing some prominent issues, e.g., external knowledge integration, multi-modal knowledge fusion, semi-supervised or unsupervised learning, etc.

In intelligent transportation area, an attention-based recurrent neural network is proposed to learn from traffic flow readings and external factors for multi-step prediction. More specifically, the attention mechanism is used to model the dynamic temporal dependencies of traffic flow data and a general fusion component is designed to incorporate the external factors. For the traffic event detection task, a multi-modal Generative Adversarial

Network (mmGAN) is designed. The proposed model contains a sensor encoder and a social encoder to learn from both traffic flow sensor data and social media data. Meanwhile, the mmGAN model is extended to a semi-supervised architecture by leveraging generative adversarial training to further learn from unlabelled data.

In social media analysis area, three deep neural models are proposed for crisis-related data classification and COVID-19 tweet analysis. We designed an adversarial training method to generate adversarial examples for image and textual social data to improve the robustness of multi-modal learning. As most social media data related to crisis or COVID-19 is not labelled, we then proposed two unsupervised text classification models on the basis of the state-of-the-art BERT model. We used the adversarial domain adaptation technique and the zero-shot learning framework to extract knowledge from a large amount of unlabeled social media data.

To demonstrate the effectiveness of our proposed solutions for smart city applications, we have collected a large amount of real-time publicly available traffic sensor data from the California department of transportation and social media data (i.e., traffic, crisis and COVID-19) from Twitter, and built a few datasets for examining prediction or classification performances. The proposed methods successfully addressed the limitations of existing approaches and outperformed the popular baseline methods on these real-world datasets. We hope the work would move the relevant research one step further in creating truly intelligence for smart cities.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Wei Wang. Being his student is my luck. His valuable advice, knowledge, and insights, as well as his constant support and patience to my research progress over the years always make me feel that there could not have been a better PhD supervisor. Wei encourages me to improve my research, to never stop thinking, and to become a independent researcher who explores his potential in different aspects. It has been a great honor to have the opportunity to work with him. This thesis would not have been completed without the support and guidance of my supervisors.

I would also like to thank my second supervisors, Prof Kaizhu Huang and Prof Frans Coenen. Their valuable help and suggestions for my PhD study motivate me to achieve higher objectives in research. It has been my privilege to have the opportunity to work with them.

I want to thank my colleagues from the School of Advanced Technology, to name a few, Yuxuan Zhao, Hang Dong, Fangyu Wu, Ziqiang Bi, Yuechun Wang, Jing Qian, Xianbin Hong, etc., who have supported me during the years as friends in the department.

Finally, and most importantly, I would like to deeply thank all my family members who have always been helping me out of difficulties and supporting me without a word of complaint. Thanks for your persistent, unconditional love, caring, and understanding. I would never be able to complete this PhD without you.

Contents

Notations

C	The classifier component in a neural network
D	The discriminator component in GAN
G	The generator component in GAN
L	Loss function for neural networks
N	The collection of instances in test set
$\|N\|$	The size of the test set
P	A learned projected matrix
Q	Knowledge graph embedding
S	A set of available labels
W	Weight matrices in neural networks
X	The collection of instances in training set
$\|X\|$	The size of the training set
Y	The collection of labels
b	Bias vectors in neural networks
f	Neural network function
h_t	The hidden state vector at the time step t
$\hat{h}_t$	The hidden state vector for a decoder at the time step t
q	Word embedding vectors
t	A time step for a time series input
u	Sentence embedding from the S-BERT model
v	Label embedding from the S-BERT model
w	A word in a sentence

w'	An adversarial example for w
x	An instance for model input
x'	An adversarial example for model input
y	Ground truth
$\hat{y}$	Network prediction
η	Perturbations to model input
θ	Parameters of a neural network model
Ω	Knowledge graph

Abbreviations

ABRNN	Attention-Based Recurrnet Neural Network
ANN	Artificial Neural Network
ARIMA	AutoRegressive Integrated Moving Average
BERT	Bidirectional Encoder Representations from Transformers
Bi-LSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
CPSS	Cyber-Physical-Social Systems
CV	Computer Vision
DBN	Deep Belief Network
DNN	Deep Neural Network
FGSM	Fast Gradient Sign Method
GAN	Generative Adversarial Network
GNN	Graph Neural Network
GRU	Gated Recurrent Unit
IoT	Internet of Things
ITS	Intelligence Transportation System
k-NN	k-Nearest Neighbour
KB	Knowledge Base
KG	Knowledge Graph
LSTM	Long Short-Term Memory
MMAT	Multi-modal Adversarial Training
mmGAN	multi-modal Generative Adversarial Network

MMN	Multi-Modal Neural Network
NLI	Natural Language Inference
NLP	Natural Language Processing
PCA	Principle Component Analysis
PeMS	Caltrans Performance Measurement System
RNN	Recurrnet Neural Network
S-BERT	Sentence-BERT
SAE	Stacked Autoencoder
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
SVR	Support Vector Regression

List of Figures

List of Tables

Chapter 1

Introduction

I go and look at a stonecutter hammering away at his rock perhaps a hundred times without as much as a crack showing in it. Yet at the hundred and first blow it will split in two, and I know it was not that blow that did it, but all that had gone before.

-Jacob August Riis

55% of the world's population lives in urban areas by 2018, a proportion that is expected to increase to 68% by 2050 [113]. The rapid urbanisation will lead to pressure for more and better infrastructure in the diminishing space available, and improved quality of life for city dwellers at a more affordable cost. Smart cities are part of the solution to the growing challenges of urbanisation, which are expected to provide profound contributions to sustainability, efficiency, and quality of services in future cities in many aspects, e.g., environment monitoring and protection, healthcare, public safety, transportation, energy management, waste proposal, etc.

The journey to smart cities goes way back to the 1970s, when Los Angeles created the first urban big data project: "A Cluster Analysis of Los Angeles". The first smart city was arguably Amsterdam with the creation of a virtual digital city in 1994. Things then speeded up in mid-2000s when IBM and Cisco launched separate initiatives. In the past few years, a number of smart cities have emerged, e.g., Amsterdam Smart City [33], Barcelona Smart City [154], Milton Keynes Smart City [79], SmartSantander [133], and etc. Different types of Internet of Things (IoT) sensors are installed in the cities to collect data. Insights gained from that data are used to manage assets, resources and services efficiently to improve the

operations across the city. Some of the notable, intelligent functionalities provided in these smart cities include public transportation, energy saving, sustainable heating, tele-care, waste recycling, traffic lighting, public open data, environment monitoring and protection, water consumption, security surveillance and tracking, to name a few.

With technologies developed in the IoT, embedded computing devices (sensors, actuators, mobiles phones and RFIDs) can be built into every fabric of urban environments and connected with each other via wireless networks; data can be preprocessed, integrated at different levels, and made available in standard formats through open services [15]. Various streaming and batch processing methods can then be used to perform simple analysis over the data to develop different smart city applications, for example, smart grid [60], smart home [157], intelligent transportation [125], and smart healthcare [176].

Review of the existing studies shows that the research community has made significant progress in data processing for smart cities in the past few years. Nevertheless, success of those existing projects under the term 'smart city' is still far away from the grand vision of the truly smart city. We need to rethink about the question "what makes a city smart?" from a more data-centric perspective, and contemplate how to exploit values and insights from the overwhelming amount of data with the state-of-the-arts technologies and interdisciplinary efforts.

Deep learning methods have continually improved the state-of-the-art in speech recognition, Computer Vision (CV) and Natural Language Processing (NLP) tasks. Recently, a number of studies have successfully applied deep learning algorithms to analyse data collected from devices or data published on the Web in domains of transportation [125, 100, 203], health care [176, 93, 105] and environmental monitoring [120, 197, 92], etc. Meanwhile, there has also been research focusing on knowledge discovery from social media data collected from Twitter, e.g. extract traffic event [37], assess disaster damage [178]. The superior performance of deep learning makes it an emerging research direction for smart city.

This thesis focuses on deep learning from smart city data. We explore the efficiency and accuracy of a number of deep learning techniques, such as Convolutional Neural Network, Recurrent Neural Network, Generative Adversarial Network, etc., for feature learning from different types of city data, such as sensor, social media and multi-modal data. Meanwhile, the designed methods will address the issues of deep learning in real-world smart city applications, i.e., external knowledge integration, model robustness, mutli-modal knowledge fusion, semi-supervised and unsupervised learning, etc.

The rest of this chapter is organised as follow. Section 1.1 discusses the scope of this thesis. Section 1.2 present the research questions . Section 1.3 shows the contributions of the thesis. Section 1.4 outlines the structure of the thesis and provides short summaries for each chapter.

1.1 Scope of the thesis

Deep learning has demonstrated superior performance over traditional machine learning methods with some notable advantages: 1) does not require a time-consuming feature engineering process, 2) learns effective representations with its deep architecture, 3) can be adapted to different types of city data. Some studies have successful applied deep learning into smart city research and achieved promising results. However, a systematic introduction to this emerging area is still limited. Therefore, the first part in this thesis is a review of the latest research on the convergence of deep learning and smart city for data processing from a technique-oriented and an application-oriented perspective.

The second part of the thesis is to demonstrate how to design models for different smart city applications. As it it neither practical nor significative to address all smart city domains, we present two smart city applications: intelligent transportation and social media analysis. Meanwhile, as city data comes from different sources with different modalities, we will show how the models could be designed to process sensor, text, and multi-modal data. While presenting these applications, we also aim to address some common challenges we face in the real-world scenario. One notable examples is that deep learning models usually require a large amount labelled data to train, while most city data is unlabelled. The thesis attempts to address this issue by leveraging techniques such as semi-supervised learning, domain adaptation, and zero-shot learning.

1.2 Research Questions

The main hypothesis in this research is:

- *A substantial amount of useful knowledge can be learned from smart city data.*

Given the large-scale, multi-modal and unlabelled characteristics of real world smart city data, the research mainly focuses on two challenging issues: (1) learning from multi-modal

data for more comprehensive prediction, and (2) learning from unlabelled data for better generalisation. Thus, two more specific hypotheses are:

- ***H1****: Data from different sources with different modalities can complement each other and be learned by deep learning models for more comprehensive prediction.*
- ***H2****: Knowledge from a large amount of unlabelled smart city data can be derived for better classification performance.*

Based on these hypotheses, the research in this thesis explores several research questions. The main question of this thesis is:

- *How to effectively extract knowledge and insights from smart city data with deep learning models?*

The question can be split into more specific questions, corresponding to the issues and challenges identified from the tasks:

- ***Q1****: How to design suitable deep neural network models for processing sensor and social media data in smart city?*

  Deep learning models (e.g., RNN, CNN, BERT, etc.) have become the state-of-the-art tools for computer vision and natural language processing tasks, but they are still premature for smart city data. It is not clear how to better process sensor data with spatial and temporal features. For example, should sensors geographically close be considered together, or should they be processed individually at each time stamp? It is also not clear how to better analyse social media data with its uncertain characteristics. For instance, How to address the noisiness, ambiguity, dynamic, and incompleteness issues of social media data?

- ***Q2****: How to design a new model to perform cross-domain knowledge discovery?*

  The nature of sensor data and social data is very different, which sensor data represents very low-level observations of our physical world, social data contains much high level semantics and directly understandable to human users. After the transformation of the raw sensor and social data, it is possible to design a new deep neural network model for multi-modal knowledge discovery. As knowledge discovery from the physical and social worlds complement and reinforce each other, the model can

perform predictive analysis which cannot be done before. Data correlations in different smart city domains can also be identified and more comprehensive knowledge can be derived. This will enable a better understanding of our city lives and reliable predictive analysis for future.

- ***Q3****: How to address deep learning limitations in real-world smart city applications?*

  Deep learning models have become unbeatable in most computer vision and natural language processing benchmarks. However, applying them to process real-world smart city data is totally different. A number of challenging issues should be properly addressed before the application is ready to use. For example, 1) City data from different sources and domains may be correlated and affect each other. How to integrate these external factors into a model should be addressed. 2) Deep learning models are easy to fool, which may lead to severe issues in real-world scenario. How to design a high quality and robust model should be considered. 3) Deep learning requires a large amount of labelled data to work, while most city data is unlabelled. How to learn from unlabelled city data is another challenge.

1.3 Contributions

The major contributions of the study can be summarised as follows.

- Our study provides a review of the latest research on the convergence of deep learning and smart city for data processing from two perspectives: while the technique-oriented review (Chapter 2) pays attention to the popular and extended deep learning models, the application-oriented review (Chapter 3) emphasises the representative application domains in smart cities. Our study showed that there are still many challenges ahead for this emerging area owing to the complex nature of deep learning and wide coverage of smart city applications. We discussed a number of future directions related to deep learning efficiency, emergent deep learning paradigms, transfer learning and knowledge fusion, and hope these would move the relevant research one step further in creating truly distributed intelligence for smart cities.

- For the traffic flow prediction task (Chapter 4), we proposed an Attention Based Recurrent Neural Network with Temporal Component (ABRNN_TC) model that considers multiple sensors' readings and external data. We leveraged an RNN Encoder-

Decoder architecture to generate a sequence of predictions representing traffic flow data in the next few minutes. Since traffic flow at different time steps may of different importance for prediction, we applied an attention mechanism to model the dynamic temporal dependencies. Meanwhile, time, weather or other external factors may significantly affect traffic flow patterns, we designed a fusion component to integrate these external factors into the model. The experimental results show that with the addition of the attention mechanism and the temporal component, the deep model can capture traffic patterns accurately and produce superior prediction results over other baseline methods.

- For the semi-supervised traffic event detection task (Section 5.1), the main contribution is the design and evaluation of a multi-modal Generative Adversarial Network (mmGAN) for the traffic event detection and classification. The proposed network attempts to address the two main limitations of existing studies on integrating data analysis of different modalities and an extremely limited amount of labelled data in big data applications. Traffic event related data comes from different sources, e.g., traffic flow data from the physical world, traffic event reports from the cyber world, traffic related tweets from the social world. To process different types of data with different representations for more comprehensive prediction, we proposed a multi-modal feature learning architecture for extracting knowledge from the sensor and social media data. Besides, since most of the real-world data is unlabelled, a particularly novel aspect of the network is the employment of semi-supervised learning based on generative adversarial training. The model has been evaluated on a large, real-world dataset, which contains 20 millions traffic flow readings and 8 millions tweets from the San Francisco Bay Area over a period of 4 months. The results confirmed that mmGAN can effectively learn useful representations characterising the multi-modal data.

- For the adversarial learning from crisis-related data task (Section 5.2), we propose a Multi-modal Adversarial Training method (MMAT). Different from most of the past studies that focused on the textual content only, the proposed MMAT model is tailored to learn useful representations from both image and text data. Deep learning models are vulnerable to adversarial examples. Small perturbations to the input will cause the models to generate incorrect results. To address this issue, adversarial training is applied to improve the robustness of the neural network against adversarial

examples. Experimental results demonstrate the proposed MMAT method is able to attain significant improvement of prediction accuracy.

- For the domain adaptation for crisis-related data task (Section 6.1), we propose a BERT-based Adversarial Domain Adaptation method (BERT-ADA). A pre-trained BERT model was utilized to extract valuable knowledge or features from tweets. However, fine-tuning the BERT model still requires labelled data which is not easy to collect especially in the early stage of a crisis. To address this issue, We applied the adversarial domain adaptation technique to BERT, which learns knowledge from a labelled source dataset and adapts it to a related unlabeled target dataset by dealing with the shift in data distributions between the source and the target domains. Experimental results demonstrate the proposed BERT-ADA model is able to gain significant improvement over other baseline models.

- For the zero-shot text classification for COVID-19 Tweet task (Section 6.2), we propose a new zero-shot learning method which exploits the use of existing knowledge graphs for the classification of large amounts of social text data (i.e. Twitter messages related to COVID-19) without training data. Following the key ideas in zero-short learning, the proposed method does not explicitly define the class labels. The pre-trained S-BERT is first used to represent tweet messages in the embedding space to be further matched with classes. As the purpose of S-BERT is to learn a sentence-level representation, while most class labels contain only one or few words, S-BERT embedding may not be as semantically consistent as word-level embedding methods. To address this problem, we construct a knowledge graph embedding model for label representation with a comprehensive knowledge graph named ConceptNet. The sentence embedding is then projected to the knowledge graph through the least-squares linear projection. The proposed model is referred to as the S-BERT-KG model. We apply the model to COVID-19 related tweet classification without any labeled data for training. Experimental results demonstrated that the proposed S-BERT-KG model is able to gain significant improvement over other baseline models and produce reasonable prediction accuracy without any labeled data.

The above contributions in this thesis have been published or accepted by peer-reviewed conferences and journals, as listed in Appendix A.

1.4 Structure of the Thesis

The thesis is organised into seven chapters. Brief summaries for each of the following chapters are presented as follows.

Chapter 2 (Deep Learning) introduces background knowledge in the area of deep learning. The introduction focuses on the basic knowledge of machine learning (training, regularisation, evaluation, etc.) and classic deep neural network models (RNN, CNN, GAN, BERT, etc.).

Chapter 3 (Smart City Data Processing) introduces background knowledge in the area of smart city data processing, which includes different city data representations (i.e., time series, Image and videos, and Text) and representative smart city applications (i.e., Transportation, Healthcare, Environment, and public safety).

Chapter 4 (Learning From Sensory Data) shows extracting knowledge from sensory data with deep neural network methods. An attention-based deep learning model is proposed for traffic flow prediction task, where external factors are embedded into the model for better prediction performance.

Chapter 5 (Learning from Multi-modal Data) presents deep learning models that processing two types of data simultaneously. Two different applications are presented in this section, which are traffic event detection (sensor and social media data) and crisis-related data classification (textual and image data). Techniques such as generative adversarial network and adversarial training are leveraged in the proposed models to further assist semi-supervised learning and robust training respectively.

Chapter 6 (Learning From Social Media Data) presents fine-grained text classification for social media data. The proposed two models are applied in real-world tasks, i.e., crisis-related data classification and COVID-19 tweet classification. As the amount of labelled social media data is limited, techniques such as domain adaptation and zero-shot learning are explored.

Chapter 7 (Conclusion and Future Work) concludes the thesis with a discussion of the extent to which the research objectives have been fulfilled. It also summaries research findings and discusses the future research directions.

Chapter 2

Deep Learning

The greatest enemy of knowledge is not ignorance, it is the illusion of knowledge.

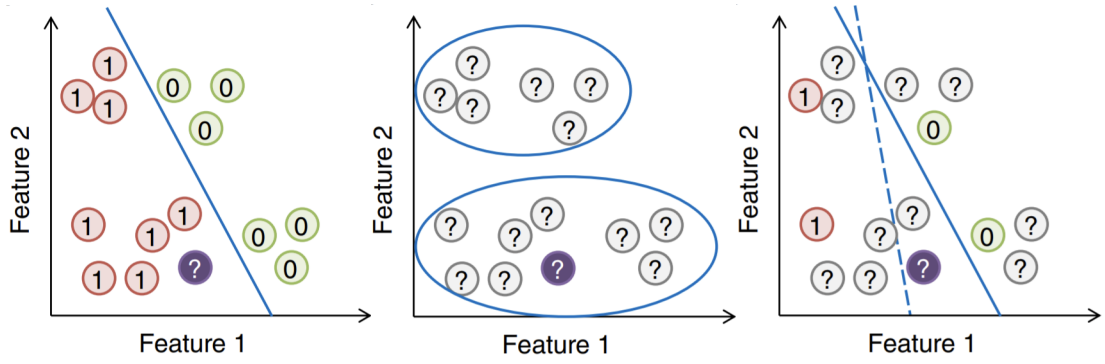
-Daniel J. Boorstin

Deep learning, as a relatively new paradigm in machine learning, has attracted substantial attention of the research community and demonstrated greater potential over traditional techniques in smart city applications. In this chapter, we give a general overview about deep learning, and then discuss in detail how different deep learning models are applied in smart city data analytics. This chapter starts off by introducing three basic types of machine learning in Section 2.1, which are supervised learning, unsupervised learning and semi-supervised learning. Then, in Section 2.2, we present the basics of training and testing process, which includes objective function, regularisation and evaluation metrics. Next, in Section 2.3, We focus on the four most widely used deep neural network models: Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Generative Adversarial Network (GAN), and Bidirectional Encoder Representations from Transformers (BERT). We show how these models could be used in smart city applications and briefly discuss how to select models for different smart city applications. Finally, we summarise the chapter and discuss some challenges and future directions of deep learning from smart city data in Section 2.4, including deep learning limitations, deep learning efficiency, transfer learning and neural architecture search.

2.1 Machine Learning

Before introducing deep learning, we cover some basics of machine learning in this section. Machine learning is the study of algorithms that improve automatically through the use of data. Give the level of availability of ground truth data, machine learning could be grouped into three categories: supervised learning, semi-supervised learning and unsupervised learning. In general, the difference of the three types could be seen from Figure 2.1 and summarised as follow.

- **Supervised learning** aims to learn a function that, given a sample of data and desired outputs, approximates a function that maps inputs to outputs.
- **Unsupervised learning** does not have any labeled outputs, its goal is to infer the natural structure present within a set of data points.
- **Semi-supervised learning** aims to label unlabeled data points using knowledge learned from a small number of labeled data points.



(a) supervised (b) unsupervised (c) semi-supervised

Figure 2.1: Supervised, unsupervised and semi-supervised learning. Figure adapted from [199]

2.1.1 Supervised Learning

Supervised learning is the process to learn a mapping function from an input variables (x) to an output variable (y).

$$y = f(x) \tag{2.1}$$

The goal is to approximate the mapping function so that given new input data (x) the model could predict the output variables (y) for that data. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process, where the algorithm iteratively makes predictions on the training data and is corrected by the teacher.

Supervised learning problems could be further grouped into regression and classification problems. The output variable for regression problem is a real value, while the output variable for classification problem is a category. For example, as shown in Figure 2.1a, the model (blue line) is learned based on the positive and negative training examples, and the unknown sample (purple circle) is classified as positive according to the model.

2.1.2 Unsupervised Learning

For unsupervised learning, algorithms mainly deal with unlabelled data and discover information from the structure of the dataset. Clustering is an important concept when it comes to unsupervised learning. It aims to find structures or patterns in a collection of uncategorized data and form natural clusters (or groups) if they exist in the data. As shown in Figure 2.1b, all examples are unlabeled, and they are grouped according to the data distribution.

2.1.3 Semi-supervised Learning

The biggest difference between supervised and unsupervised machine learning is the availability of labelled training data. Acquiring a large amount of labelled data for supervised learning is an expensive process, since most real world data are unlabelled. Unsupervised learning algorithms, trained on unlabeled data, usually have limited application spectrum. To counter these disadvantages, the concept of semi-supervised learning was introduced. It is trained upon a combination of labeled and unlabeled data, which contain a very small amount of labeled data and a very large amount of unlabeled data. It allows the model to use unsupervised learning techniques to discover and learn the structure in the input variables, and use supervised learning techniques to make predictions for the unlabeled data with knowledge learned from labelled samples as shown in Figure 2.1c.

2.2 Training and Testing

Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. This learning process is called training. The quality of the models can be evaluated through the testing process with test data. In this section, some key features are introduced, which are objective function, regularisation and evaluation metrics.

2.2.1 Objective function

An objective function is normally used to measure the error (or distance) between the predicted output and the desired output. A common objective function for classification tasks is Cross Entropy (CE) as shown in Equation 2.2, where x is an instance in the training set, $p(x)$ is the true probability distribution of the dependent variable while $q(x)$ is the predicted probability distribution.

$$CE = -\sum_{x} p(x) log q(x) \tag{2.2}$$

Common objective functions for regression tasks are Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE) as defined in Equation 2.3, where X is the training set, $|X|$ is the size of the training set, y is the ground truth and $\hat{y}$ is the network output for regression problem.

$$\begin{aligned} \text{MSE} &= \frac{1}{|X|}\sum_{x}(y(x) - \hat{y}(x))^2 \\ \text{RMSE} &= \sqrt{\frac{1}{|X|}\sum_{x}(y(x) - \hat{y}(x))^2} \\ \text{MAE} &= \frac{1}{|X|}\sum_{x}|y(x) - \hat{y}(x)| \end{aligned} \tag{2.3}$$

Some deep learning models can also be trained in unsupervised ways in which the output tries to recover the input and the objective is to minimise the reconstruction. For example, the work in [66] showed that deep belief networks can be trained in an unsupervised manner (pre-training), followed by a supervised fine-tuning, which resulted in superior performance.

After determining objective function, deep models are then trained to minimise the objective function with backpropagation and gradient descent techniques. Backpropagation algorithm distributes the error computed at the output layer backwards and the gradients of weights at different layers are calculated. At each layer, the gradient descent algorithm computes a gradient vector, and adjusts the weight vector along the opposite direction of the gradient vector to minimise the objective function. In practice, gradient-based learning algorithms, e.g., Stochastic Gradient Descent (SGD) [141], Adam [81] and RMSprop [171], have been widely adopted together with backpropagation for neural network training.

2.2.2 Regularisation

One of the common goals of machine learning algorithms is to generalise to unseen data. However, when a model is trained for too long or when the model is too complex, it can start to learn the noise or irrelevant information in the training set. When the model learns the noise and fits too closely to the training set, the model becomes "overfitted" and it is unable to generalize well to unseen data. Regularisation is a critical instrument in preventing overfitting. Some of the most common regularisation techniques for deep learning are: dataset augmentation, L1 and L2 regularisation, early stopping, and dropout, as suggested in [53].

Dataset augmentation - overfitting can be a common problem when training set is too small compared with the number of model parameters to be learned. While an existing dataset may be limited, for some applications one may create synthetic data through a number of operations, e.g., rotate, scale and inject random unrelated images to enlarge a dataset. Besides creating synthetic data, multi-task learning and transfer learning techniques are also commonly used. In multi-task learning, related tasks using different datasets can be learned simultaneously. The work in [85, 158, 74] applied multi-task learning to jointly learn people's movement and transportation mode patterns. Often when a training dataset may be too specific or small to learn a good model from scratch, transfer learning can be applied by pre-training a model with large available dataset and then fine-tuning the model with the data for specific tasks such as fire detection [109], parking lot detection [174], and crisis-related tweets classification [115].

L1 and L2 regularisation - to prevent model from becoming too complex (e.g., large weights) and learning all the details and noise in the training dataset, a regularisation term can be added to the objective function. L1 regularisation (or Lasso regression) and

L2 regularisation (or Ridge regression) are commonly used not only in deep learning but also many other machine learning algorithms. L1 regularisation is defined using absolute values of the weights and can perform some sort of feature selection, while L2 regularisation is defined using the squared values of the weights to penalise large model parameters.

Early stopping - in an ideal situation, as a model sees more data both training and test errors should constantly decrease. However, after a certain number of epochs, the model may start to overfit and learn noise in the training set. In this case, the training error keeps going down while the test error starts to increase. Early stopping is used here to find the right moment to stop training to minimise the test error.

Dropout - it refers to a strategy that randomly drops out some units (hidden and visible) in a deep neural network to make nodes become more insensitive to the weights of the other nodes. It provides a way of approximately combining many different neural network architectures efficiently [161].

2.2.3 Evaluation Metrics

Evaluation metrics are used to measure the quality of machine learning models. For regression tasks, MSE, RMSE and MAE defined in Equation 2.3 are commonly used to assess model performance. For classification tasks, five widely used metrics are Accuracy, Precision, Recall, F-measure and Hamming loss as shown below, where N denotes instances in the testing data, $|N|$ denotes its size, $\hat{y}_i$ and y_i denote the predicted and actual labels for the ith instance respectively.

- Accuracy (A), defined as the fraction of the correctly predicted labels to the total number of labels presented (union of predicted and actual ones), computed as

$$\mathrm{A} = \frac{1}{|N|}\sum_{i=1}^{N}\frac{|\hat{y}_i \cap y_i|}{|\hat{y}_i \cup y_i|}$$

- Precision (P), defined as the fraction of the correctly predicted labels to all the predicted labels, computed as

$$\mathrm{P} = \frac{1}{|N|}\sum_{i=1}^{N}\frac{|\hat{y}_i \cap y_i|}{|\hat{y}_i|}$$

- Recall (R), defined as the fraction of the correctly predicted labels to all the actual labels, computed as

$$\mathrm{R} = \frac{1}{|N|} \sum_{i=1}^{N} \frac{|\hat{y}_i \cap y_i|}{|y_i|}$$

- F-measure (F_1), defined as the harmonic mean between precision and recall, computed as

$$\mathrm{F}_1 = \frac{2PR}{P+R}$$

- Hamming loss (H) measures the number of misclassified labels, computed as

$$\mathrm{H} = \frac{1}{|N| \cdot |S|} \sum_{i=1}^{N} \sum_{j=1}^{S} |\hat{y}_i \oplus y_i|$$

  where $\oplus$ is the "Exclusive or" operator that returns zero when the target and prediction are identical and one otherwise and S is the total number of available classes. The lower the value, the better the performance.

2.3 Deep Neural Network

Machine learning methods, e.g., artificial neural networks, fuzzy systems, and evolutionary computation, have achieved remarkable results in modelling, learning, searching, and optimisation problems for smart city applications [61, 173, 145]. Deep learning, a relatively "young" learning paradigm in the machine learning family, has in fact its origin from Artificial Neural Network (ANN). Simply speaking, when a neural network contains more than one hidden layer, it is considered as a 'deep' architecture. A gradient descent algorithm called backpropagation (BP) was proposed in 1980s to train neural network and has played an important role in neural network training since then. However, one of the major reasons that ANN with multiple fully connected layers have not gained popularity in many real-world applications for decades is the computation complexity. In 2006, a breakthrough research first showed that training Deep Neural Network (DNN) in an unsupervised manner (pre-training), followed by a supervised fine-tuning, could result in good performance. In 2012, the research group led by Hinton won the ImageNet competition by using convolutional neural network that almost halve the classification error rate. Since then, the study of deep learning has achieved a series of milestones in various domains and

has been applied to smart city research.

The idea of deep learning, also inspired by biological processes, has made very deep models computationally feasible for real-world applications. For example, in a convolutional network, the connectivity between neurons resembles the organisation of neurons in animal visual cortex [88]. Each cortical neuron only responds to stimuli within a limited region of a visual field (also known as the receptive field); in a recurrent network, weights are shared among layers which not only reduces the number of parameters to be learned but also generalises better for input sequences of different lengths [88]. Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction, and is able to discover intricate structures from natural data in its raw forms without the needs of sophisticated feature engineering and tuning [88].

Compared to traditional machine learning methods, deep learning can model extremely sophisticated functions through layers of non-linear transformation trainable from the beginning to the end. Methods based on deep models have significantly improved the state-of-the-art in a wide array of problems, such as natural language processing (or more specifically, neural machine translation) [166, 31, 17], computer vision [86, 153, 64], and speech recognition [65, 57, 10]. The characteristics of deep learning also make it attractive for analysing smart city data which has complex nature, i.e., different modalities (streaming, time-series, image, video and text), large amount (continuous data generated by millions of sensing devices), spatial and temporal dependency, etc. Researchers in smart cities have applied deep learning in many domains, e.g., traffic prediction, healthcare, air quality prediction and public safety.

In this section, we introduce the most widely used deep neural network models: Recurrent Neural Network, Convolutional Neural Network, Generative Adversarial Network, Bidirectional Encoder Representations from Transformers, and some hybrid models.

2.3.1 Recurrent Neural Network

A Recurrnet Neural Network (RNN) [46] contains links among neurons, and after unfolding it forms a directed graph along a sequence. This allows RNN to process data that can be modelled as temporal sequences of variable lengths, $x = (x_1, ..., x_T)$. At each time step t, the hidden state h_t of the RNN is updated using $h_t = f(h_{t-1}, x_t)$, where f is a non-linear function, which can be as simple as an sigmoid function and as complex as a long short-

term memory unit. RNNs use the internal states to capture dependency among input data in a sequence, which makes them suitable to tasks such as natural language processing, speech recognition and smart city applications in which data demonstrates strong temporal correlations. As vanilla RNNs suffers from various limitations, they have not been used in any real-world applications. In what follows, we present two important RNN units: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), and two widely used architectures: Bi-directional RNN and RNN encoder-decoder; and then discuss their usage in different smart city applications.

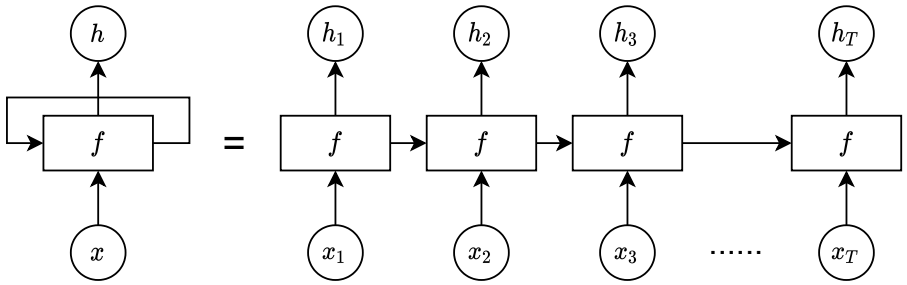


Figure 2.2: Recurrent Neural Network (RNN) Architecture

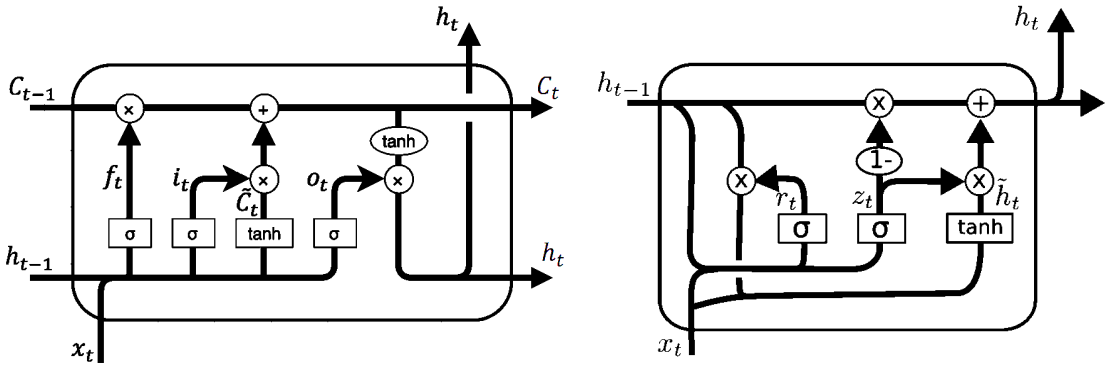
Long Short-Term Memory

Vanilla RNNs have difficulties in modelling long sequences as the gradients in parameter updates tend to either explode or vanish during backpropagation. Long Short-Term Memory (LSTM) has been proposed to solve the vanishing and exploding gradient problem by introducing the idea of Constant Error Carousels (CEC) [67]. In the original LSTM, the activation function of the unit is replaced by the identity function in the CEC to enforce constant error flow. Later, various extended models have been proposed, e.g., by adding forget gate and peephole connection [57] in order to address the limitations of the original LSTM [67].

We briefly introduce the LSTM unit, following the notations used in [57, 118]. As shown in Figure 2.3a, it contains a cell C, an input gate i, an output gate o and a forget gate f. The subscript t represents a particular time step. A standard LSTM updates the hidden state h by iterating the following steps shown in Equation 2.4, where all the W and U matrices are the learnable weights and the b vector represents the bias term (We ignore the subscripts for simplicity).

$$
\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
C_t &= f_t * C_{t-1} + i_t * tanh(W_C x_t + U_C h_{t-1} + b_C) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
h_t &= o_t * tanh(C_t)
\end{aligned} \tag{2.4}
$$

The standard LSTM based RNNs have been applied in many smart city applications, e.g., traffic prediction [170, 101], healthcare analysis [93] and air quality prediction [120]. Taking traffic prediction as an example, the task is to predict traffic flow x^i_{T+1} at time step $T+1$ based on the historical traffic flow sequence $X = \{x^i_t | i = 1, 2, ..., n, t = 1, 2, ..., T\}$, where n denotes the number of traffic observation stations. At each time step, the traffic flow vector x^i_t is fed into the corresponding input layer. After training, the network learns the relationship of different stations and time steps, and uses the last h_t to predict the traffic flow of all stations at time $T+1$.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 2.3: Two widely used recurrent units in RNNs. Diagrams adapted from [118]

Gated Recurrent Unit

Gated Recurrent Unit (GRU), as shown in Figure 2.3b, is another RNN unit introduced by Cho *et al.* [31]. It contains two gates: update gate z and reset gate r, where the update

gate helps the model determine how much of the past information to remember and the reset gate is used to decide how much of the past information to forget. The hidden state h is then updated iteratively using the following procedure shown in Equation 2.5, where all the W and U matrices are the learnable weights and the b vector represents the bias terms.

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * \sigma_h(W_h x_t + \\ &\qquad U_h(r_t * h_{t-1}) + b_h) \end{aligned} \tag{2.5}$$

The GRU unit is simpler and more efficient than the standard LSTM. It has become increasingly popular in natural language processing [31], speech recognition [10], traffic prediction [48] and air quality prediction [3].

Bi-directional RNN

A standard RNN can learn representations from data from previous time steps; however, representations from future time steps may help better understand the context and eliminate ambiguity. For example, in handwriting recognition, the performance can be significantly enhanced if the letters located before and after the current letter were known. Bi-directional RNNs [149] was proposed by stacking two LSTM RNNs, one processing the sequence from left to right, the other one from the opposite direction, and finally concatenating the output of the two RNNs. With this structure, the output layer can integrate information from both past and future states.

Bi-directional RNNs have been commonly used in natural language processing [165] and speech recognition [57]. For sensor data processing, recent studies applied Bidirectional Long Short-Term Memory (Bi-LSTM) to recognise human activities [110, 44]. In this task, the input is a discrete sequence of equally spaced samples $\{x_1, x_2, ..., x_t\}$, where each data point x_t is a vector of individual samples observed by sensors at time t. The samples are segmented into windows of maximum time T and fed into the network with one direction from time 1 to T and another direction from time T to 1. The network can output the probabilities of different activity labels after a softmax layer. Both work [110, 44] reported the state-of-the-art performance compared to conventional techniques.

RNN Encoder-Decoder

In some applications, the input and output sequences have different lengths, e.g., in machine translation, the input sentence and the desired target sentence usually have different lengths. An important and effective technique for such application is the RNN based encoder-decoder architecture [31]. It contains two RNNs, one learns to encode an input sequence of certain length into a context vector representation (the encoder) and the other learns to decode the context vector representation back into an output sequence of different length (the decoder).

This architecture allows smart city applications to produce a sequence of predictions for time series data. For example, in the traffic flow prediction task, an encoder can be used to learn the representation of the historical traffic flow data and a decoder can be used to generate a sequence of predictions representing traffic flow in the next few minutes or even hours. However, the RNN Encoder-Decoder architecture still has some issues not considered while applying to traffic flow prediction. For example, (1) traffic flow at different time steps may be of different importance for prediction, (2) time, weather or other external factors may significantly change traffic flow patterns. To address these issues, we proposed an attention-based RNN model for the traffic flow prediction task in Chapter 4. The attention mechanism is used to model the dynamic temporal dependencies of traffic flow data and a general fusion component is designed to incorporate the external factors.

2.3.2 Convolutional Neural Network

A typical Convolutional Neural Network (CNN) [87] usually consists of a number of convolutional layers, pooling layers, and fully connected layers as its hidden layers as illustrated in Figure 2.4. The convolutional layer aims to learn filters that represent features of the input (e.g., a particular shape) and generate a feature map. The pooling layer performs non-linear down sampling, which combines a cluster of neurons at one layer into a single neuron in the next based on non-linear functions such as max pooling and average pooling. Then, fully connected layers are added on the top of convolutional and pooling layers for final output. In Figure 2.4, the output is represented by $p(y|x)$, which stands for the probability distribution over the dependent variable y, given an input x. In recent years, CNN has achieved superior results in image and video recognition [64], speech recognition [2], and natural language processing [50].

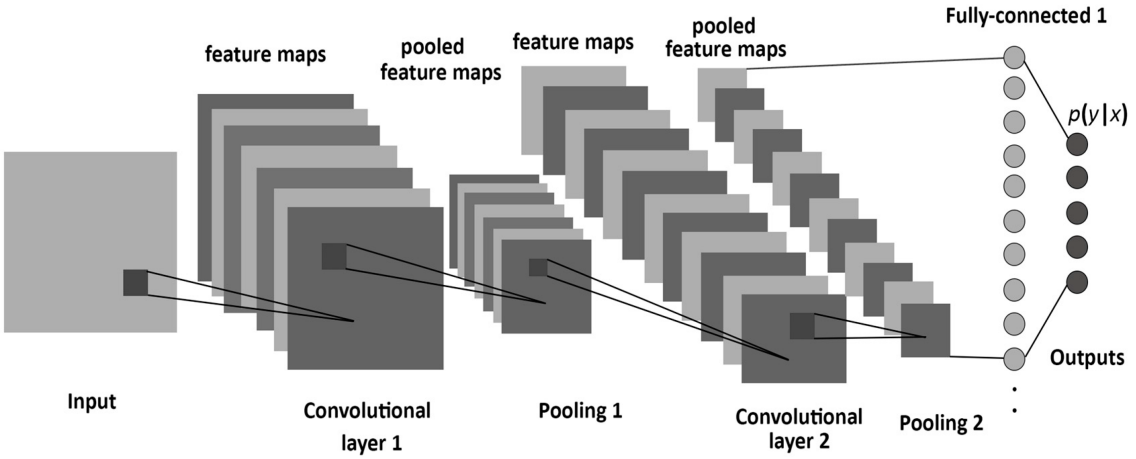


Figure 2.4: A typical Convolutional Neural Network architecture. Figure adapted from [87]

In smart city research, CNN has been applied in applications related to healthcare, public safety and transportation. In processing sensor data, CNN can make use of spatially correlated sensor signals for tasks such as human mobility [203] and human activity recognition [184]. As CNN is primarily designed for processing images and videos, input adaptation, which transforms the sensor data to virtual images, is necessary. The study in [203] first partitioned a city into regions and then calculated the inflow and outflow of crowds in each region based on data collected from GPS trajectories and bike trips. Each region represents a pixel in a virtual image; and inflow and outflow of crowds represent the value for that pixel. For human activity recognition tasks, most sensors produce multi-dimensional 1D readings such as acceleration signal. There are mainly two types of transformation: the first is to treat each dimension as a channel, and perform 1D convolution, and the second is to combine readings from multiple sensors to form a virtual 2D image and then perform 2D convolution [184].

For image and video recognition tasks, many configurations and variants have been proposed in the literature for smart city applications. In [102], a random CNN was proposed to solve the vehicle license plate recognition problem. It has four layers implementing filter bank convolution, rectified-linear activation, spatial pooling, and divisive normalisation, respectively. The final architecture was learned from thousands of random possibilities. CNN has also been applied to provide robust face descriptors, and achieved state-of-art accuracy in face verification and recognition tasks [140, 192].

Recently, some notable variants of the CNN for image classification tasks, e.g., AlexNet [86], VGG [153], GoogLeNet [167] and ResNet [64], have been developed. When use these large scale networks for smart city applications, e.g., parking lot detection [9], fire detection [109], human mobility [203] and vehicle detection [97], a model pre-trained with large scale image datasets and fine-tuned with dataset in particular domain is usually preferred.

2.3.3 Generative Adversarial Network

Generative Adversarial Network (GAN), proposed in 2014, is initially designed to generate realistic images given a random signal. The fundamental idea of GANs is to set up an adversarial game between a discriminator and a generator. The goal of the discriminator is to distinguish whether a sample is drawn from the true data or generated by the generator. On the contrary, the generator is optimised to produce samples that are not distinguishable by the discriminator. In this way, the generator and discriminator compete with each other to boost their performance in a seamless manner. The architecture of GAN is shown in Figure 2.5.

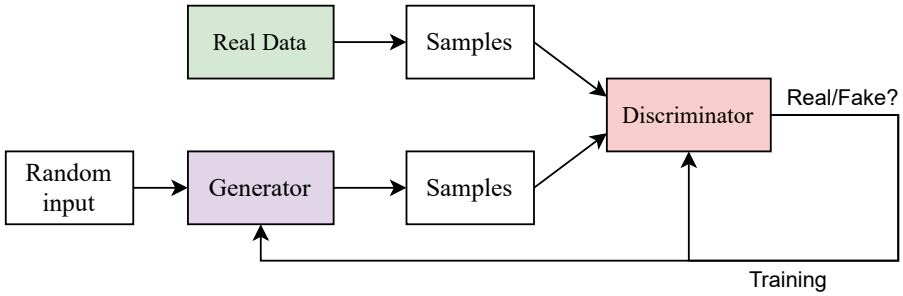


Figure 2.5: Generative Adversarial Network (GAN) Architecture

The discriminator outputs a value $D(x)$ indicating the chance that x is a real image. The objective is to maximize the chance to recognize real images as real and generated images as fake. To measure the loss, we use cross-entropy as described in Section 2.2.1: $plog(q)$. So the objective becomes:

$$\max_D V(D) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}\,(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.6}$$

On the generator side, its objective function wants the model to generate images with the highest possible value of $D(x)$ to fool the discriminator:

$$\min_G V(G) = \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.7}$$

In summary, the D and the G play a two-player minimax game (described in Equation 2.8), where G wants to minimize V while D wants to maximize it.

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.8}$$

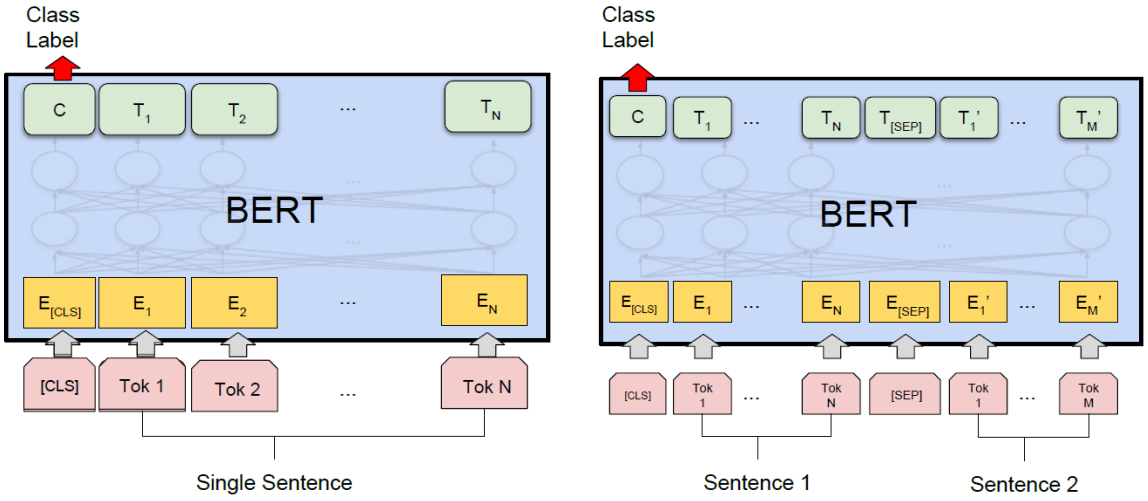
The GAN framework is not restricted to image generation tasks. In fact, recent studies have applied GAN in smart city applications. For instance, the work in [29] applied GAN in traffic data imputation task for intelligent transportation systems; The work in [32] leveraged GAN to produce high quality synthetic patient records to protect personal sensitive medical information.

Besides, GAN can be further extended to semi-supervised architecture by adding samples from the generator G in GAN to the dataset and labelling them with a new class, "generated", denoted as $y = K + 1$ [160, 146, 38, 212]. On the basis of this idea, we proposed a multi-modal Generative Adversarial Network (mmGAN) in Section 5.1 for semi-supervised traffic event detection. As detecting the "generated" samples and classifying the rest labelled samples are two different tasks, we designed a discriminator and a classifier with a few shared layers and result in better classification performance. Besides, since GAN is mainly used for processing image data, we further designed a multi-modal architecture in the proposed model to learn from traffic sensor data and social media textual data.

2.3.4 Bidirectional Encoder Representations from Transformers

Pre-trained word embeddings, e.g., word2vec [104] and GloVe [127], as an important component of modern NLP tasks, can offer significant improvements over embeddings learned from scratch. These context-free models generate a single "word embedding" representation for each word in the vocabulary, although the meaning of such words may differ in different scenarios. On the contrary, contextual models, i.e., OpenAI GPT [138], ELMo [131], ULMFit [71] and Bidirectional Encoder Representations from Transformers (BERT) [42], generate a representation of each word based on other words in a sentence. These methods of pre-training contextual models on a large network with a large amount of

unlabeled data and fine-tuning in downstream tasks has made a breakthrough in several NLP tasks. Different from OpenAI GPT and ELMo models, BERT contains deeply bidirectional transformer layers, where each word is contextualized using the words to its left or right. This architecture allows BERT to achieve state-of-the-art performance in NLP tasks, e.g., text classification, question answering and language inference. The architecture of fune-tuning BERT on classification tasks is shown in Figure 2.6.



(a) Single Sentence Classification Task

(b) Sentence Pair Classification Task

Figure 2.6: Illustrations of Fine-tuning BERT on Classification Tasks. Figure adapted from [42].

While using the pre-trained BERT, a tokenizer is applied to split sentence input into tokens $\{Tok_1, Tok_2, ..., Tok_n\}$, add special tokens (e.g., [CLS], [SEP]) and convert these tokens into indexes of the tokenizer vocabulary. The first token of the sequence is [CLS] which contains the special classification embedding. The output hidden state corresponding to this token C is used as the aggregated sequence representation for classification tasks. The [SEP] is used to separate two sentences while the input to the models is a sentence pair.

Some recent studies that leverage BERT model in smart city applications can be found, e.g., clinical NLP tasks [8] and crisis events classification [1]. When BERT are applied in domain-specific tasks, fine-tuning with a small amount of labelled data is still required. As most real-world data is unlabelled, we further explore how to better utilise the pre-trained

BERT model, knowledge from related tasks, or external knowledge base for unsupervised learning tasks in Chapter 6. On the basis of BERT, we leveraged the adversarial domain adaptation technique and the zero-shot learning framework to extract knowledge from a large amount of unlabelled social media data and achieved promising classification results.

2.3.5 Hybrid approaches

Hybrid model is referred to as the combination of more than one type of deep learning models. One particularly interesting hybrid model is the combination of CNN and RNN, which enhances the capabilities of the deep learning model in processing data that exhibits strong spatial and temporal correlations. Such data in fact is very common in many smart city applications. In [183], the authors proposed a geo-convolution component to integrate geographic information into the classical convolution to capture the spatial dependency of taxi trajectories. A recurrent layer was then added on the top of the geo-convolution to capture temporal dependencies among the trajectories for accurate travel time estimation. In [106], a 3D-CNN was used to extract local spatial-temporal features from a video. Then, these features were fed into to a recurrent network for global temporal modelling for dynamic hand gestures detection, which cannot be easily realised with a standalone CNN or RNN. The work also reported that the model outperformed competing state-of-the-art algorithms based on evaluation with several standard benchmark datasets.

2.3.6 Model Selection for Smart City Data

One deep model can be applied to solve different problems in the smart city domain, and one specific problem can be addressed by several deep learning models or combination of them. One of the questions to be asked is how to select the best model for a given city data type. To answer the question, we summarise the applications of different deep models, together with the input representation and the representative references in Table 2.1. Today, CNN has been the dominate model in processing data with strong spatial correlation, e.g., images used in medical research [68, 142, 215] and public surveillance [97, 156], as well as geographic location data used in human mobility [184]. RNN has been the first choice for modelling sequence data with strong temporal dependencies, e.g., sensor time series of traffic flow or speed [170, 101, 48], air quality [120, 3, 92], human activity [110, 44], etc. This can be roughly used as a first guideline in selecting an appropriate model. Nevertheless, there have been studies showing that CNN could also model temporal

Model	Input representation	City data	Reference
RNN	Time sequence data $\{x_1, x_2, ..., x_t\}$	traffic flow or speed (sensor)	[170, 101, 48]
		air quality data (sensor)	[120, 3, 92]
		healthcare data (sensor)	[93]
		human activity (sensor)	[110, 44]
CNN	2D or 3D matrix $\begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix}$ vector $[x_1, x_2, ..., x_n]$	human mobility (sensor)	[184]
		human activity (sensor)	[203]
		parking lot image	[174, 9]
		medical image	[68, 142, 215]
		disaster image	[178, 109]
		twitter data (text)	[115]
		face image	[164, 148]
		vehicle image	[97, 156]
GAN	2D or 3D matrix	traffic flow (sensor)	[29]
		electronic health record	[32]
		face image	[172]
BERT	Sentence $[w_1, w_2, ..., w_n]$	crisis-related tweet (text)	[1]
		clinical text	[8]
CNN+ RNN	Sequence of 2D or 3D matrices	taxi trajectories (sensor)	[183]
		hand gesture video	[106]
		meteorology data (sensor)	[193]

Table 2.1: Overview of deep learning models for different type of city data.

dependency in human mobility recognition [203] and short text classification [115]. Many types of city data demonstrate spatio-temporal correlations, e.g., GPS trajectories, videos, and meteorology data. Hybrid models that combine CNN with RNN [183, 106, 193] (or with some spatial temporal transformation or embedding mechanisms) work well and tend to perform better than any single models. Besides, some other deep learning models (e.g., Stacked Autoencoder (SAE) and Deep Belief Network (DBN)) not widely used recently will not be introduced in this thesis.

Technically, it is difficult to make a straightforward conclusion on what the best model is for a particular problem. There are at least two reasons for this: (1) the lack of large-scale experiments and benchmark studies. Our study showed that in the experiments in most of the research work, the results were only compared to those generated by either conventional machine learning techniques or deep models falling in the same category with some parameter changes or minor architecture modifications. We have not found any

notable studies performing horizontal comparison, i.e., evaluation against deep models in other categories; and (2) the inherent complex nature of deep learning. Usually, there are extremely large number of parameters to train and performance of a model is sometimes very sensitive to parameter settings. Such a tedious process often prevents one from finding the best model and settings for a specific task. We expect to see that more benchmark datasets will be shared by the research community for more comprehensive evaluation studies.

2.4 Summary and Discussion

In this chapter, we gave a general overview about deep learning, and then discussed in detail how different deep learning models are applied in smart city data analytics. This chapter started off by introducing three basic types of machine learning in Section 2.1, which are supervised learning, unsupervised learning and semi-supervised learning. Then, in Section 2.2, we present the basics of training and testing process, which includes objective function, regularisation and evaluation metrics. Next, in Section 2.3, We focused on the four most widely used deep neural network models: Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Generative Adversarial Network (GAN), and Bidirectional Encoder Representations from Transformers (BERT). We showed how these models could be used in smart city applications and briefly discuss how to select models for different smart city applications.

Research on deep leaning from smart city data is still in an early stage given the complex natural of city data. Although deep learning has demonstrated great success and outperformed many other conventional machine learning methods, it is not a versatile solution for every applications in smart city. There are at least a number of limitations that smart city researchers have to take into account: 1) deep learning models are notoriously expensive to train, and it is not uncommon for the training to take days even with high-end computing devices. Meanwhile, it is also difficult to configure the models; sometimes many experiments need to be conducted in order to find proper or near optimal configurations; 2) for supervised learning, large amount of training data is needed. In applications where only data of small or medium size is available, performance of a deep learning model may not be guaranteed; and 3) a learned deep model is usually hard to interpret in intuitive ways. It usually works like a “black box” that maps an input to an output, and may not be a good option for applications in which users are interested in knowing how decisions are

made. Therefore, some potential research directions for deep learning from smart city data can be identified, e.g., efficient models for capability-constrained devices, transfer learning with limited labelled data, neural architecture search for automate machine learning, etc.

The primary concern in most studies has been high accuracy (e.g., for classification) or low error (e.g., for prediction). Computation complexity, e.g., training cost, has been largely ignored. It also seems that the researchers have not been fully aware of some latest development in deep learning. As network architectures become deeper and deeper, learning tends to be extremely hard and inefficient. Much of the recent research emphasises on learning efficiency, for example, Deep Residual Network (ResNet) in [64] introduced shortcut connections that allow convolution neural networks to have super deep structures without significantly increasing computation complexity. Transformer based methods [177] used only attention mechanisms with position embeddings to learn representations from sequence data. It not only improved the state-of-the-art in the area of machine translation but also achieved signification improvement in training time. Likewise, we hope that smart city researchers pay attention to the learning efficiency in designing deep models and include computation complexity as the additional evaluation metric.

Transfer learning is a learning architecture that gained increasing attentions recently. In practical applications, very often the labelled datasets may be too specific or small to train effective models from scratch. In this case transfer learning can be applied: first learning an model on a large related dataset and then fine-tuning the network with the data for the specific task. In a sense, transfer learning encourages reuse of existing learned models and has the potential to save considerable computation cost. In fact, it has been applied in some studies such as fire detection [109] and parking lot detection [174]. Other transfer learning techniques, e.g., domain adaptation and zero-shot learning (will be introduced in Chapter 6), allow model to learn from related data resources and make prediction without any labelled data for training. In the future, we would expect extensive applications of transfer learning in many other smart city domains.

Developing effective deep models requires substantial efforts in architecture engineering which is time consuming and expensive process. Neural Architecture Search (NAS) is an emerging deep learning paradigm that aims to automatically find or design effective deep architectures for specific tasks minimising human participation. The interesting research direction has quickly drawn considerable attention of researchers from computational intelligence, in particular, evolutionary computing. Recently, evolutionary algorithms (e.g., genetic programming) [163, 162] and reinforcement learning based methods [216, 18] have

been used to design new neural architectures. Research has reported that the deep models found by NAS can outperform a large number of manually designed models. Although the current NAS methods mostly focus on image classification tasks, we believe that the paradigm will be applied to other deep models and used for the smart city domains in the near future.

Chapter 3

Smart City Data Processing

Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom.

-Clifford Stoll

Smart city is a multi-disciplinary research field by its own nature and many of its applications have close connections with other long-existing research areas such as transportation, mobile computing and healthcare. An important reason that such applications are often described as being "intelligent" or "smart" is that they apply artificial intelligence techniques to perform sophisticated data analytics to extract meaningful and useful insights and understand the cities better.

In this chapter, we give a general overview about smart city data processing. This chapter starts off by introducing the background of smart city and the framework of smart city data processing in Section 3.1. Then, in Section 3.2, we present the city data representation: time series, image and video, and text. Next, in Section 3.3, We focus on the four representative application domains for which deep learning methods have been applied and demonstrated notable performance improvement, i.e., Transportation, Healthcare, Environment and Public safety. Finally, we summarise the chapter and discuss some future directions of smart city applications in Section 3.4.

3.1 Background of Smart City

It is reported by the United Nations that 55% of the world population resided in urban areas in the year of 2018 and by 2050 the figure is expected to increase to 68% [113]. Continuous urbanisation poses severe challenges on sustainable development and living quality of urban residents. The vision of smart city is to make more efficient uses of scarce resources, and improve quality of citizen lives and public services [200]. Numerous smart city applications have been developed and deployed, for example intelligent transportation [125, 100, 203], smart healthcare [176, 93, 105], environment monitoring [120, 197, 92], and public safety [97, 169, 11], just to name a few. Data has become the most ssential ingredient in smart city applications. Review of the existing studies shows that the research community has made significant progress in data processing for smart cities in the past few years. Nevertheless, success of those existing projects under the term 'smart city' is still far away from the grand vision of the truly smart city.

From a data-centric perspective, the key concept in smart cities lies in the sophisticated data analytics for understanding, monitoring, regulating and planning the city [84]. It is widely accepted that the process of smart city data analysis can be abstracted as four layers although different work may have some minor variations [100, 203, 92, 85]: data acquisition, data preprocessing, data analysis, and service provision, as shown in Figure 3.1: (1) the data acquisition layer is concerned about collecting and storing smart city data from various domains and sources; (2) the data preprocessing layer is responsible for preprocessing (e.g., data cleaning, selection and interpolation) to obtain data of higher quality before analytics, as smart city data of different modalities often contains noise, uncertainty and missing values; (3) the data analytics layer is to perform intelligent analysis using various machine learning techniques to discover previously unknown knowledge and insights specific to different applications, e.g., classification models can be used to recognise human activities and regression models to predict traffic flows; and (4) the service provision layer is to develop intelligent services and applications based on the outcome of data analytics, for example, with the patterns and events detected from traffic, pollution and weather sensory data analysis to provide better public services.

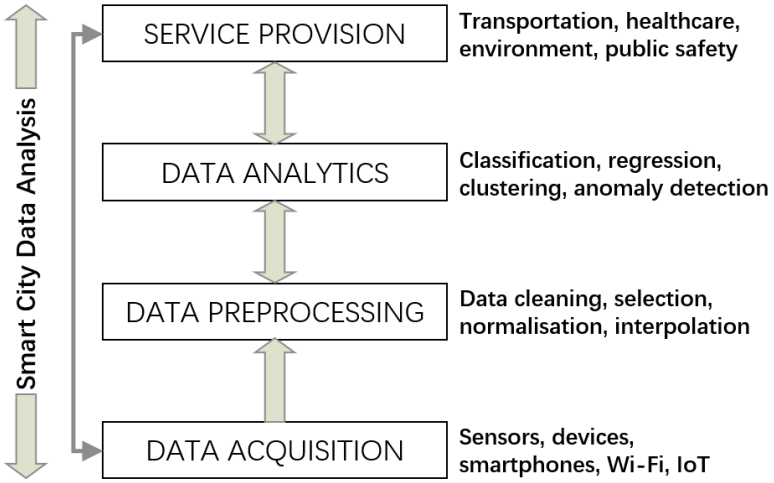


Figure 3.1: A general framework for smart city data analysis.

3.2 Data representation

City data can be collected from a large number of heterogeneous sources, e.g., sensor data from fixed and mobile sites as well as user-contributed data in participatory sensing [41]. From the many specific city data types as shown in Tables 2.1, 3.1, 3.2, 3.3 and 3.4, three representative data modalities can be observed: sensor data, image/video data and text data. These data modalities and their representations suitable for deep learning are described below.

3.2.1 Time series

Different sensing devices have been deployed in many city applications, e.g., traditional physical sensors (fixed and mobile) have been widely used to measure physical phenomenon (e.g., temperature, air quality, light and traffic flow); wearable sensors and smartphones equipped with accelerometers, magnetometer, and gyroscopes, have been used to recognise human activities and behaviour. Sensors in different applications have very different sampling rates: while Electroencephalogram (EEG) sensors used in healthcare offer millisecond-range temporal resolution, air quality sensors report in minutes. Sensor measurement data is almost always associated with metadata, e.g., spatial information (where the measurement happens) and temporal information (when), which is important for sensor data representation and analysis.

Sensor Data Representation - There have been various representation formalisms proposed for sensor data, e.g., the Semantic Sensor Network (SSN) ontology [34], FUTS data model (Frequently Updated, Timestamped and Structured) [137], sensor stream representation [19] and the mobile sensor data model [214]. These representations focus on individual sensor observations and highlight the importance of modelling temporal and spatial information. With deep learning, input sensor data can be roughly represented as a temporal sequence of elements in the form $x = x_1, x_2, ..., x_T$, where x denotes the sequence to be processed and T denotes the length of x. Each x_t in the sequence is a vector representing the data at time step t. Spatial information may be embedded in x_t and its meaning is application dependent. For example, in traffic flow prediction [170, 101] and human activity recognition with smartphones [184, 110], x represents a sequence of sensor values (traffic flow/speed and accelerometers/gyroscopes, respectively). In these applications, spatial information does not necessarily need to be considered as the sensors are installed at fixed locations. However, in processing GPS sensor data, spatial information has to be considered and can be embedded in x through Geo-convolutional operations [183].

3.2.2 Image and videos

Image and video data have been pervasive in smart cities and can be collected in many ways: surveillance cameras are widely installed in city areas for public safety and criminal investigation; Unmanned Aerial Vehicles (UAVs) equipped with remotely controllable devices (i.e., sensors, cameras, and actuators) [108] are used for crowd surveillance [107] and disaster management [178]; satellite images are analysed to detect and classify objects on earth for environment protection and weather forecasting; citizens use smartphones to take images and videos to record daily lives and report social events.

Image and Video Data Representation - A digital image is usually represented as a three-dimensional matrix with numeric values. Each point in an image is called a pixel, denoting the colour value at that specific position. The number of rows and columns of an image is fixed and indicates the resolution of the image. To effectively characterise an image and make its representation robust to common variances, e.g, translation, illumination and occlusion, choice of features is crucial. Methods for image features can be roughly divided into two categories: hand-crafted and machine learned features. For hand-crafted features, one needs to manually design a feature extraction pipeline and algorithms based on expert

knowledge. The most representative handcrafted features include Scale-Invariant Feature Transform (SIFT) [99] and Histogram of Oriented Gradients (HOG) [188]. In contrast, with deep learning, features can be extracted automatically from image datasets. For image-based video representation, the standard method computes the representation by pooling all the descriptors from all the frames in a video. It can be summarised as: (1) obtain the descriptors for individual frames; (2) apply normalisation on frame descriptors; (3) perform average pooling on frame descriptors to obtain video representation, i.e., $x_{video} = \frac{1}{N}\sum_{i=1}^{N} x_i$, where x_i is the frame-level descriptor and N is the total number of frames extracted from the video; and (4) apply re-normalisation on the video representation.

3.2.3 Text

As smartphones and wireless networks become pervasive, the idea of 'citizen sensing' has been proposed to describe the activities of network of people who actively observe, collect, analyse, report and disseminate information through text, audio, or video [151]. While data produced by the citizen sensors differs semantically from sensing data generated by the physical devices to a great extent, it is a necessary complementary data source for many smart city applications. One of the typical applications is to analyse short textual messages posted on social media platforms (e.g., Twitter) to discover knowledge from multiple perspectives in real time, e.g., social event detection in Twitris [111], traffic event detection [39, 12], disease activity tracking [7], and natural disease monitoring [35, 115]. One key advantage of making use of social textual data is that the discovered knowledge has a higher level of semantics which is more intuitive for human beings to comprehend.

Text Data Representation - Processing textual data heavily relies on natural language processing techniques. In deep learning, textual data can be represented as temporal sequence of variable lengths, similar to sensor data representation, i.e., $x = \{x_1, x_2, ...x_t, ..., x_T\}$, where x denotes a sentence or text block and x_t a vector representing a word at time step t. Two common word representation methods are the one-hot representation and word embedding [103, 104]. One-hot representation of a word sets the corresponding element as '1' ('0' for all others) in a vector whose dimensionality equals to the size of the vocabulary. The representation is simple and not able to capture relational structure of the lexicon. The word embedding representation has become the norm for natural language processing tasks using deep learning. It involves a mathematical embedding from a space with one dimension per word to a continuous vector space with a much lower dimension. The resul-

tant embedding has a key characteristic that similar words end up having values closer to each other although each word is assigned with a different vector [103, 104].

3.3 Applications

Conventional techniques used in smart city applications have at least three major limitations: (1) they usually need a complex and time-consuming feature engineering process before perform data analysis. This has been reported in many existing studies, e.g., parking space detection [191], medical image analysis [112] and face recognition [28]; (2) conventional techniques with shallow architectures, e.g., ANN with one or few hidden layers, may not be able to learn effective representations, which often results in poor performance. Research in different domains, e.g., traffic flow prediction [26], medical diagnosis [4] and air quality prediction [180], has pointed out the same issue; (3) they cannot directly process raw smart city data which are often noisy and contain many missing values. They only work with high-quality datasets and cannot generalise well with corrupted or missing data as shown in [11, 95, 30]. In contrast, deep learning models can learn a robust representation from corrupted input, e.g. corrupted sensor data [45], incomplete electronic health records [20] and masked face images [91].

As reviewing an exhausted list of these applications is neither possible nor significative, we only study four representative application domains for which deep learning methods have been applied and demonstrated notable performance improvement, i.e., Transportation, Healthcare, Environment and Public safety. A classification of these application areas is shown in Figure 3.2. Information about the nature of the studies, datasets used and methods is summarised in Tables 3.1, 3.2, 3.3, and 3.4, respectively. The "Remarks" columns in the tables highlight the most distinctive features of each individual studies.

3.3.1 Transportation

With deep learning, intelligence transportation systems are able to discover knowledge from traffic data and enable users make safer, more coordinated, and smarter use of the transport networks. We focus on three applications: transport flow, human mobility, and parking applications. Features of the selected studies are highlighted in Table 3.1.

Reference	Application	Data	Method	Remarks
Huang *et al.* [74] and Koesdwiady *et al.* [85]	Transport flow	Real time traffic data in California (PeMS dataset) and entrance-exit station data of a highway	DBN	Traffic flow prediction; [74] incorporated multitask learning in DBN; [85] combined weather parameters to improve prediction performance.
Tian *et al.*[170] and Ma *et al.* [101]	Transport flow	Real time traffic data in California (PeMS dataset) and 42387 records of travel speed data in Beijing	RNN	Traffic flow/speed prediction; used LSTM to model long-term dependencies.
Chen *et al.* [29]	Transport flow	Real time traffic data in California (PeMS dataset)	GAN	Traffic flow imputation; using parallel data and GAN to enhance traffic data imputation.
Song *et al.* [158]	Human mobility	1.6 million GPS records in Japan and transportation network data (i.e. road structure and POI information)	RNN	City-wide human mobility prediction; used multi-task LSTM to jointly learn people movement and transportation mode patterns.
Zhang *et al.* [203]	Human mobility	GPS trajectories data, bike trip data, weather conditions and events	CNN	Forecasts the inflow and outflow of crowds in each region of a city; used ResNet to model spatial correlation; collected timestamp data to model temporal correlation.
Xu *et al.* [194]	Human mobility	600 millions taxi trip data contains GPS location and timestamp of pick-up and drop-off event in NYC	RNN	Taxi demand prediction; predicted future taxi demand in each area of a city based on the recent demand and other relevant information, e.g., weather, time, and drop-offs.
Wang *et al.* [183]	Human mobility	2 billion GPS trajectories with time, driver ID and weather condition	Hybrid	Travel time estimation; introduced a CNN-RNN hybrid approach to model both spatial and temporal information.
Amato *et al.* [9] and Valipour *et al.* [174]	Parking	CNRPark and PKLot datasets contain images of parking lots and segmented parking spaces	CNN	Parking lot occupancy detection; applied AlexNet and VGGNet, respectively.

Table 3.1: Overview of research using deep learning for transportation related applications.

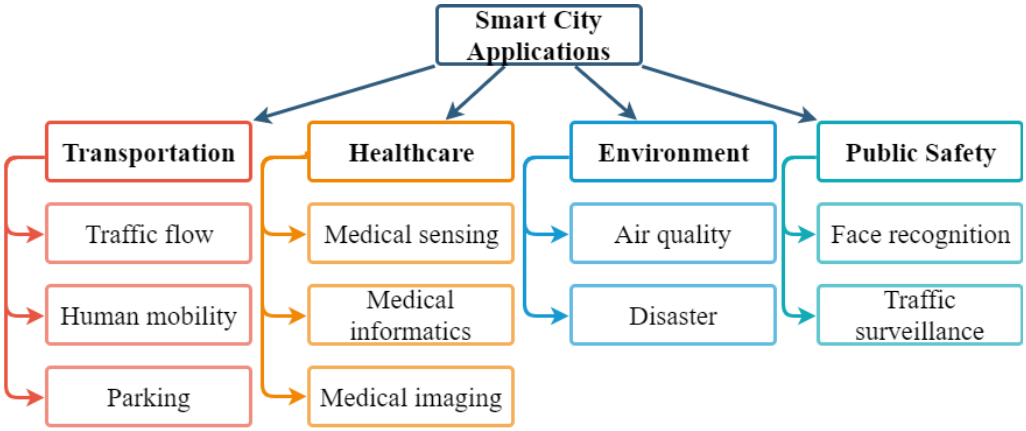


Figure 3.2: Classification of the deep learning research in smart city data applications.

Transport flow

Road traffic information is needed by individual travellers, business sectors, and government agencies to make better travel decisions, alleviate traffic congestion, and improve traffic operation efficiency [205]. Loop sensors have been widely embedded in pairs in major roads (e.g., highways) to collect traffic data for transportation analysis, i.e., detecting traffic flow/speed by counting the number of traversing vehicles and the time interval that vehicles travel across a pair of sensors. Data collected from different roads can be used in combination to predict traffic flow/speed, analyse traffic congestion, and make better urban transportation planning.

A number of conventional models have been used for traffic flow prediction in the literature, e.g., AutoRegressive Integrated Moving Average (ARIMA), k-Nearest Neighbour (k-NN) [27], Support Vector Regression (SVR) [77] and ANN [26]. For time-series based methods like ARIMA, a linear method is often used and the future traffic flow is only predicted based on historical traffic flows on a particular road. The methods ignore the fact that transportation system is a highly correlated network. For k-NN and SVR, a complex feature engineering is mostly needed, which requires prior knowledge of the transportation domain. Also without proper pre-processing and cleaning, they cannot easily work with missing data. As mentioned earlier, ANN with shallow architectures often results in poor performance.

As the loop sensors are deployed at fixed locations, spatial information is usually not used in traffic flow and speed prediction. It is natural to see that only RNN, SAE and DBN have been used in existing research. The work of Huang *et al.* [74] was the first study to apply deep learning for the traffic flow prediction problem by using DBN to learn effective features in an unsupervised fashion. In addition, it added a multitask regression layer on the top of DBN for supervised prediction and reported around 5% improvements over the state of the art. As traffic is usually affected by other factors such as weather condition, the work in [85] developed a DBN based deep learning model to fuse traffic data and weather data for more accurate prediction. As DBN cannot model temporal dependency in the data, RNN has also been used for traffic flow/speed prediction [170, 101]. Both work used LSTM based RNN for short-term traffic flow/speed prediction and reported better performance over other deep learning models. For the traffic flow imputation task, the work in [29] showed a novel approach using parallel data and GAN to enhance traffic data imputation, which outperform conventional methods such as SVR and SAE.

Human Mobility

Location acquisition technologies like GPS and WiFi enable people to record a sequence of timestamped location history called trajectories. The trajectory data has been used for individual movement prediction or urban transportation modelling. However, most of studies only focused on single or multiple road segments, rather than the dynamics of human mobility on a city-wide scale [158, 203]. Recently, deep learning research made use of trajectory data for many purposes, e.g., human mobility prediction, taxi demand analysis, and estimate travel time.

Predicting human mobility in a city is of great importance for traffic management and public safety [211]. The work in [158] described an intelligent transportation system for simulating and predicting human mobility and transportation mode at a citywide level. The study collected GPS records of about 1.6 million anonymous users throughout Japan from 2010 to 2013, and transportation network data (i.e., road structures and point of interests information) of many cities in Japan. It applied LSTM based RNN with a multi-task learning architecture to jointly learn human mobility and transportation mode. It was used to predict future movement of citizens and their transportation modes in a large-scale transportation network. The work in [203] proposed the use of ResNet for crowd flow prediction. It first partitioned a city into regions based on the longitude and latitude, and

then applied ResNet to forecast inflow and outflow of crowds in each region of a city based on data from different sources, such as GPS trajectories, bike trips, weather conditions and events.

Predicting taxi demand can help better organise taxi fleets and minimise wait-time for passengers and drivers. The work in [194] proposed a LSTM based RNN model for taxi demand prediction in each areas of a city, based on the data collected from around 600 millions taxi trips during 2013 and 2016 at the New York City. Future taxi demand was predicted based on not only the recent demand but also other relevant information, e.g., weather, time and pick-up and drop-off events. GPS trajectories also can be used to estimate travel time in a city. The work in [183] proposed a hybrid deep learning framework (CNN + RNN) to estimate travel time on any route in a city. CNN was used to capture the spatial dependency of taxi trajectories and RNN was used to model the temporal dependencies among these trajectories. It collected about 2 billion GPS trajectories together with the corresponding time information and driver IDs in Beijing and Chengdu. The travel time of both the entire path or each local path was then estimated based on the GPS trajectories, weather, time and driver information using the hybrid model.

Parking

As the number of vehicles in cities keeps increasing, finding a parking space efficiently has become a problem for drivers. The idea of smart parking has been introduced to optimise parking space usage, improve the efficiency of parking operations, alleviate traffic congestion, and reduce CO emissions. Vacant parking space detection using only visual information is still an open problem. Many conventional techniques are only tailored and fine-tuned to specific and ideal contexts and scenarios [201, 191]; therefore, they cannot be easily generalised, or be adapted to a different parking lot. In contrast, with deep learning models, especially CNN, the result is robust to disturbance created by partial occlusions, presence of shadows and variation of light conditions.

The work in [9] proposed a decentralised and efficient solution for visual parking lot occupancy detection based on an AlexNet. The system was deployed at the parking lot of the research campus in Pisa as part of the smart city application. Similarly, the work in [174] presented a robust parking lot detection algorithm based on a light version of VGGNet. It was used to report the occupancy status of a parking stall based on the

images taken by cameras. A fully functional system including server-side image analysis and frontend user interface was developed as a smart parking application.

3.3.2 Healthcare

In recent years, many smart healthcare applications have been developed to improve traditional healthcare systems and provide better support for prevention, diagnosis, and treatment of disease. Due to the large number of diseases presented in the existing research we categorised the review based on the nature of medical data: medical sensing, medical informatics, and medical imaging.

Medical Sensing

Medical sensing data collected from pervasive sensors, such as smartphone, wearable, and ambient sensors, has been used to develop many different types of applications, ranging from health condition assessment, infant/elderly/patient care to disease diagnosis. As accurate estimation of food intake and daily energy consumption can help solve obesity and improve personal health, the work in [132] proposed a smartphone-based, assistive system to help patients and doctors control diet-related health conditions. The system allowed users to take images of food and then used CNN to classify the food images and automatically calculate the amount of calorie intake.

Healthcare of infant, elderly and people with disabilities or chronic diseases can be improved with medical sensing data and human activity recognition techniques. In [72], an intelligent surveillance system was developed to recognise human activities such as falling to floor and baby crawling. A three stream CNN was proposed for activity recognition, in which the spatial stream CNN was trained for human detection; a temporal stream CNN was trained for activity recognition; and a moving stream net was used to detect speed and direction of the movement. Recognition of hand gestures (e.g., sign language interpretation) is important for people with hearing disabilities. In [106], a recurrent 3d convolutional neural network was proposed to perform simultaneous detection and classification of dynamic hand gestures. Some diseases are related to the change of certain body movements such as Parkinson disease; and wearable sensors can be used to detect these movements. In [63], an assessment system based on DBN is developed to differentiate Parkinson disease states with data collected from 34 people with Parkinson disease in naturalistic settings.

Reference	Application	Data	Method	Remarks
Pouladzadeh *et al.* [132]	Medical sensing	10000 high-resolution food images	CNN	Food calorie estimation.
Huang *et al.* [72]	Medical sensing	UCF101 and HMDB-51 datasets about elderly and children care	CNN	Recognised human actions such as falling to floor and baby crawling; a three stream convolution neural network was proposed.
molchanov *et al.* [106]	Medical sensing	multimodal dynamic hand gesture dataset captured with depth, color and stereo-IR sensors	Hybrid	Hand gestures detection and classification; online system without noticeable lag; a recurrent 3D CNN was proposed.
Lipton *et al.* [93]	Medical informatics	Anonymised clinical time series extracted from the EHR system at Children's Hospital LA	RNN	Multilabel classification of diagnoses; included drop out, target replication and auxiliary output.
Choi *et al.* [32]	Medical informatics	Two datasets from PAMF and a MIMIC-III dataset	GAN	Patient records generation; proposed a medGAN model to generate high-dimensional discrete variables (e.g., binary and count features).
Alsentzer *et al.* [8]	Medical informatics	clinical text from the approximately 2 million notes in the MIMIC-III dataset	BERT	Clinical NLP tasks; provide a clinical BERT Embedding for medical NLP tasks.
Hoo *et al.* [68]	Medical imaging	Public available thoracoabdominal lymph node datasets and interstitial lung disease dataset	CNN	Anomaly detection; Compared architectures for detecting interstitial disease and lymph nodes.
Ronneberger *et al.* [142]	Medical imaging	Datasets from EM segmentation challenge and ISBI cell tracking challenge	CNN	Cell segmentation; U-Net with deformation augmentation.
Zhou *et al.* [215]	Medical imaging	C3H10T1/2 and C2C12 Datasets from the CMU cell image analysis group	CNN	Cell mitosis detection; 3D convolution kernel for processing consecutive image frames.

Table 3.2: Overview of research using deep learning for healthcare related applications.

Medical Informatics

Medical Informatics helps enhance healthcare quality by analysing large amount of aggregated and cumulated medical data (mainly textual). Electronic health record (EHR) is a type of such data containing well-structured, textual information about patients, such as diagnosis, medical history, medications and allergies, immunisation records, vital signs and laboratory test results. Efficient mining of EHR can provide valuable insight into disease management. The work in [93] applied RNN with LSTM units to classify diagnosis (formulated as a multi-label classification task) based on Pediatric Intensive Care Unit (PICU) time series data (e.g., body temperature, heart rate, blood pressure, and others). The work in [32] proposed a medGAN model to generate high-dimensional discrete variables (e.g., binary and count features) via a combination of an autoencoder and GAN, which could achieve comparable performance to real data on many experiments including distribution statistics, predictive modeling tasks and a medical expert review. As there is no pre-trained model in the clinical domain, the work in [8] explored and released BERT models for clinical text, and showed that using a domain-specific model yileds performance improvements on common clinical NLP tasks as compared to nonspecific embeddings.

Medical Imaging

Medical imaging is an important technique in creating visual representations of the interior of human bodies to facilitate doctors in disease diagnosis and treatment. Automated image analysis has become mandatory to modern healthcare because of the amount of imaging data. Research in this area heavily relies on techniques developed for image processing. It is not surprising to see that deep learning, especially CNN based methods, have been adopted rapidly due to their superior performance in common tasks, e.g., CT abnormality detection [68], cell segmentation [142], cell mitosis detection [215], foot ulcer classification [56], interstitial lung disease classification [14], and left ventricle segmentation [16]. Many other medical imaging tasks can also be found in applications for detecting and diagnosing neuronal, ophthalmic, pulmonary, and cardiac diseases. We refer the readers to [94] which provides a comprehensive overview of deep learning techniques for medical imaging from both an application perspective and methodology driven perspective.

3.3.3 Environment

Rapid global urbanisation has posed great threats (e.g., air pollution, extreme weather, wild fire and earthquake) to the environment that we live in. Effective monitoring and protection of environment has been one of most significant tasks for smart cities. Through continuous analysis of the environment related data, e.g., meteorological sensing data, remote sensing images and even social media data, we can have accurate monitoring and forecasting, and identify causes for the problems; more importantly, issue pre-warning, better organise resources for rescue and avoid catastrophic effects in natural disasters. We focus on the research related to air quality and disaster (man-made and natural).

Air Quality

It is reported by the National Oceanic and Atmospheric Administration (NOAA) that the U.S. spends tens of billions of dollars each year to reduce air pollution in order to protect public health and environment [117]. However, poor air quality is still the main cause for cardiovascular and respiratory diseases which results in tens of thousands deaths across the U.S. annually. Accurate prediction of air quality is a challenging task as it is affected by multiple factors such as meteorology, traffic, location, time and social events. Taking into consideration different factors, the work in [197] proposed a DNN (deep neural network with fusion components) based approach to predict the Air Quality Index (AQI), which is a widely used metric to indicate how polluted the air is. A spatial transformation component is used to address spatial correlation and a distributed fusion network is used to merge all the influential factors. The work in [120] aimed to predict PM2.5 concentration for 52 cities in Japan. It used the historical PM 2.5 concentrations along with other features (e.g., wind speed and rain precipitations) to compute the PM2.5 concentration levels several hours ahead. In their work, the deep model is pre-trained with an autoencoder specifically designed for time series data, and fine-tuned using RNN. The results showed that the method outperformed VENUS, the office PM2.5 prediction system developed by the National Institute for Environmental Studies in Japan. To include dynamic spatio-temporal correlations and external factors, the study in [92] presented a RNN model with two components: a multi-level attention mechanism to model the dynamic spatio-temporal dependencies and a general fusion module to incorporate the external factors from different domains, and outperform other conventional models in real-world air quality and water quality prediction tasks.

Reference	Application	Data	Method	Remarks
Yi *et al.* [197]	Air quality	Hourly air pollutants, meteorological data and weather forecast data in China	DNN	Air quality prediction; Hand-crafted spatial transformation component to address spatial correlation; fusion network to fuse different factors.
Ong *et al.* [120]	Air quality	PM2.5 prediction system called VENUS developed by the National Institute for Environmental Studies in Japan	Hybrid	PM2.5 concentration prediction; pre-training with auto-encoder; use RNN to model time series.
Liang *et al.* [92]	Air quality and water quality	real-time water quality information and meteorological readings in China	RNN	Air quality and water quality prediction; use multi-level attention-based recurrent neural network.
Vetrivel *et al.* [178]	Disaster	Two groups of datasets based on multi-view oblique images from manned aircrafts and UAVs	CNN	Disaster damage detection; a multiple-kernel-learning framework combines CNN features and 3D point cloud features.
Muhammad *et al.* [109]	Disaster	A dataset of 68457 images collected from different fire datasets of both images and videos	CNN	Fire detection in indoor and outdoor environments; a prioritisation mechanism to change the priority of camera nodes; dynamic channel selection algorithm.
Nguyen *et al.* [115]	Disaster	Labelled twitter datasets: CrisisNLP, CrisisLex, and AIDR	CNN	Crisis-Related tweets classification with pre-trained word embeddings.
Abavisani *et al.* [1]	Disaster	Multimodal crisis dataset: CrisisMMD	hybrid	Crisis event classification; multimodal fusion: BERT for textual data and CNN for image data.

Table 3.3: Overview of research using deep learning for environment related applications.

Disaster

Images collected from remote sensing devices and surveillance cameras are important data sources for disaster analysis. In [178] the authors designed a multiple kernel learning framework based on features extracted from CNN and 3D point clouds. The method was used to detect severe building damages caused by destructive disasters (e.g., earthquakes) from oblique aerial images taken by manned aircrafts and UAVs. The work in [109] proposed an early fire detection system that can detect fire in both indoor and outdoor environments. It used a pre-trained Alexnet for real time fire detection based on images captured from surveillance cameras.

Social media data can complement data collected from physical devices in many aspects. In times of crisis, people may use social media platforms to publish situational updates, find useful information, and seek for help. Efficient and accurate analysis can help city authorities gain situational awareness of the affected citizens, and identify critical infrastructure damage and medical emergencies. The work in [115] presented such an application to identify disaster events from relevant tweets during crisis. A CNN based framework was adopted to capture the salient n-gram information by convolution and pooling operations. Results showed that the deep learning based method substantially outperformed the conventional supervised learning methods (e.g., Support Vector Machines, Random Forest, and Logistic Regression) with current and previous event (out-of-event data) data. The work in [1] present a new multimodal fusion method (BERT+CNN) that leverages both images and texts of tweets as input for crisis-related event classification, which outperform standalone BERT and CNN models.

3.3.4 Public safety

Smart city also aims to provide safer environments for citizens. With the installation of numerous surveillance cameras and the generated image and video data, enforcement of relevant regulations, criminal investigation and transportation safety can be significantly improved. Research in this aspect heavily replies on techniques developed for image/video processing, e.g., object recognition and identification. This is also one of the reasons that CNN based framework has been extensively used as illustrated in Table 3.4. We focus on the two important applications of face recognition and traffic surveillance.

Reference	Application	Data	Method	Remarks
Sun *et al.* [164]	Face recognition	Labeled Faces in the Wild (LFW)	CNN	Face verification; a set of high-level feature representations learned through deep learning, referred to as Deep hidden IDentity features (DeepID).
Schroff *et al.* [148]	Face recognition	Labeled Faces in the Wild (LFW) and YouTube Faces DB	CNN	Face verification, recognition and clustering; using CNN to directly optimise the embedding, rather than an intermediate bottleneck layer as in conventional CNNs.
Menotti *et al.* [102]	Traffic surveillance	The dataset was obtained from Brazilian license plate images captured in a real-world setting	CNN	Vehicle license plate recognition; Using two networks, one for recognition of digits and the other for letters; was the first time that a random CNN was used for vehicle identification.
Shivakumara *et al.* [152]	Traffic surveillance	20105 images for experimentation in this work, 18270 images from UCSD dataset, 1835 images from MIMOS dataset	Hybrid	Vehicle license plate recognition; combined merits of both CNN and RNN to handle issues of poor quality, complex background, blur and noise.
Sochor *et al.* [156]	Traffic surveillance	Boxcars dataset with the 3D bounding boxes and contains 21250 vehicles (63750 images) of 27 different makes.	CNN	Vehicle detection and recognition; collected and annotated a new dataset BoxCars; showed that additional information easily obtainable in real time from static surveillance cameras can boost the CNN verification performance greatly.
Liu *et al.* [97]	Traffic surveillance	VeRi-776 dataset contains about 40000 images of 619 vehicles captured by 20 surveillance cameras.	CNN	Vehicle re-identification; collected the dataset VeRi-776; proposed a deep learning-based, progressive vehicle re-identification approach for urban surveillance.

Table 3.4: Overview of research using deep learning for public safety related applications.

Face Recognition

Human face conveying identity of a person plays an important role in social interactions. Automated face recognition techniques have been developed for a variety of applications requiring efficient and "smart" interactions with different environments such as rooms, cars, and office desks [128]. It is the process of uniquely verifying (1:1 match) or identifying (1:N match) a person by comparing and analysing patterns based on the person's facial contours.

Many conventional methods for face verification represent faces with over-complete low-level features and employ shallow models, for example, in [24], each face image was encoded into 26K learning-based (LE) descriptors, and then the L2 distance between the LE descriptors was calculated after performing Principle Component Analysis (PCA). Chen *et al.* [28] extracted 100K LBP descriptors at dense facial landmarks with multiple scales and used Joint Bayesian for verification after PCA. Recent research showed that CNN has been effective in automatically extracting high-level visual features and significantly outperformed conventional methods in face recognition.

To achieve ultimate accuracy, most of the CNN models tend to be deeper or multiple local facial patch ensemble, which result in prolonged training time and waste of storage space. To alleviate the problem, Sun *et al.* [164] proposed a light CNN framework to learn a compact embedding for face representation. Features extracted from different face regions were complementary and further boosted the performance. It achieved 97.45% face verification accuracy on Labeled Faces in the Wild (LFW) dataset, while only requiring weakly aligned faces. In [148] Schroff *et al.* presented a unified system called FaceNet for face verification (is this the same person), recognition (who is this person) and clustering (find common people among these faces). The method was based on learning a Euclidean embedding per image using CNN, which was trained such that the squared L2 distances in the embedding space directly correspond to face similarity: faces of the same person have small distances and faces of different people have large distances. CNN have also been applied to face age estimation system for access control and surveillance monitoring tasks, e.g., preventing children from entering unauthorised or dangerous zones, preventing teenagers under legal age from buying cigarette from vending machines [123].

Traffic surveillance

Traffic surveillance cameras have been widely deployed in city roads, generating numerous images and videos for vehicle detection, tracking, classification and verification [196],

which can be used in many applications, such as traffic law enforcement, electronic payment systems, automatic toll collection and traffic monitoring systems. Many conventional methods are based on detection of the license plate for Region of Interest (ROI) extraction [95, 30, 11]. Unfortunately, in many occasions license plate of a vehicle may not be discovered because the plate might be missing, cloned or simply unrecognisable.

Automatic vehicle detection and recognition aims to process videos recorded from stationary cameras over roads (e.g. CCTV cameras installed nearby traffic intersections and junctions) and then transmit them to the surveillance centre for recording and processing. Sochor *et al.* [156] presented an automatic vehicle detection and recognition method which saves a lot of time and effort for users trying to identify blacklisted vehicles from a large surveillance image database. This work showed that additional information easily obtainable in real time from static surveillance cameras can boost the CNN verification performance greatly. Recently, the application of vehicle re-identification (re-ID) has gained considerable research attention. Liu *et al.* [97] proposed a vehicle re-ID system to quickly discover, locate and track the target vehicles across multiple cameras. It employed CNN to extract appearance attributes as a coarse filter, and Siamese neural network to accurately verify vehicles based on the license plate. The work reported 9.28% improvement over the state-of-the-art methods in terms of mean average precision.

3.4 Summary and Discussion

In this chapter, we gave a general overview about smart city data processing. This chapter started off by introducing the background of smart city and the framework of smart city data processing in Section 3.1. Then, in Section 3.2, we present the city data representation: time series, image and video, and text. In Section 3.3, We focused on the four representative application domains for which deep learning methods have been applied and demonstrated notable performance improvement, i.e., Transportation, Healthcare, Environment and Public safety.

While smart city applications promise to make our living environment safer, greener and more sustainable, they also bring a number of challenges issues in terms of knowledge discovery.

Smart city is a Cyber-Physical-Social Systems (CPSS) and the data from the cyber, physical and social worlds has different characteristics, for example, trustworthiness of the data collected from the social world may be questionable; and data from the physical world

usually has low-level semantics and cannot be easily integrated with social data. However, knowledge discovered from different worlds can complement and reinforce each other in order to derive more meaningful insights, e.g., traffic anomaly can be identified by both loop sensors installed on roads and social media data published online. The information from the two different sources varies in nature but provides details complementing each other. We believe that more insightful knowledge can be derived if we can discover the correlations among data from the physical, cyber, and social worlds. This knowledge fusion process requires techniques which can effectively handle cross-domain and multi-modal data. If we were to think a smart city as an integrative system we have to interrelate the outcome of data analytics in each individual domains and fuse the learned knowledge wherever possible to create a system that is truly smart. Deep learning techniques are considered as good candidates [69, 187] for the knowledge fusion.

The research in [186] proposed the idea of "distributed intelligence", which transfers much of the intelligent computation (e.g., data aggregation, smoothing, reasoning and analytics) from data centres to the much smaller while autonomous units at different levels in a city environment, e.g., sensor nodes, gateways of Wireless Sensor networks, smartphones, or mobile edge networks, in order to tackle the challenges brought by the volume and velocity of big city data. However, we have to take into consideration the capability of these computing devices as they are far less powerful compared to servers in data centres. For deep learning, we envisage the need to develop more lightweight models so they can be implemented on the capability-constrained devices. The idea of distributed intelligence also provides great potential for transfer learning, where pre-trained models are implanted into distributed devices for fine-tuning.

As it it neither practical nor significative to address all the issues for deep learning from smart city data, we present two smart city applications (intelligent transportation and crisis-related tweet analysis) with different types of data (sensor, text, and multi-modal data) in the next three chapters to address different issues, i.e., 1) external factors integration, 2) robust model training, 3) multi-modal feature learning, 4) semi-supervised learning, 5) unsupervised learning.

Chapter 4

Learning From Sensory Data

The search for truth is more precious than its possession.

-Albert Einstein

With the development of Internet of Things (IoT), embedded devices, e.g., sensors, actuators, mobiles phones and RFIDs, can be built into every fabric of urban environments and connected with each other. Sensory data generated by these devices can be preprocessed, integrated, and made available in standard formats through open services [15]. Many machine learning techniques, e.g. classification, regression and clustering methods, have been applied to process and analyse IoT data to extract useful knowledge. Real-world applications have been developed and deployed to help citizens better understand their surroundings and informs city authorities to provide better and more efficient public services.

A number of smart city applications have been developed from sensory data analysis, e.g., intelligent transportation [100], smart healthcare [93], and environment monitoring [136]. As it is impractical to show how to use deep learning in all these areas, we select traffic flow prediction as one typical task of sensory data processing in this chapter. In Section 4.1, we introduce the traffic flow prediction task and propose an attention based recurrent nerual network for better prediction performance. We summarise the chapter and discusses some future directions in Section 4.2.

4.1 Traffic Flow Prediction

Traffic flow information is crucial for individual travelers, business sectors, and government agencies to make better travel decisions, alleviate traffic congestion, and improve traffic operation efficiency [205]. With the rapid development and deployment of Intelligence Transportation System (ITS), traffic flow prediction has gained increasing attention in recent years. With the widespread traffic sensors, available traffic data (e.g., loop sensors, GPS, cameras, social media, etc.) for analysis is exploding; with advanced networking technologies, big traffic data can be efficiently and securely collected, processed, cached, shared and delivered. Through sophisticated analysis of historical and real-time traffic data, ITSs enable users to make safer, more coordinated, and smarter use of the transport networks.

Over the past few years, deep learning methods have been applied to capture traffic patterns and provide traffic predictions even without prior knowledge in transportation areas, and have shown superior results over traditional models (e.g., ARIMA, k-nn, SVR and ANN) as discussed in Section 3.3.1. However, these studies still leave some issues unaddressed. For example, existing studies focus on predicting traffic flow at next time step (e.g. in 15 minutes), while travelers may need a sequence of traffic flow predictions (e.g., traffic situation in the next few hours) in order to make better, longer-term travel decisions. Meanwhile, external information (e.g., time of the day, day of the week, national holiday) of traffic flow is usually ignored, e.g., weekdays, weekends and national holidays should be considered separately. To address the above limitations, we introduce an Attention-Based Recurrent Neural Network architecture with Temporal Component (ABRNN_TC) for traffic flow prediction. The proposed model is compared to other baseline methods and shows superior performance. To further illustrate the use of the method, we present a case study in real-time traffic event detection.

In Section 4.1.1, we presents the details of the proposed attention-based recurrent neural network architecture for traffic flow prediction. Section 4.1.2 shows the experimental and evaluation results.

4.1.1 Attention-Based Recurrent Neural Network

As has been discussed in Section 2.3.1 and 3.3.1, RNN can be directly used to process traffic sensor data given its ability in processing data represented as temporal sequences. However, using RNN for traffic flow prediction have three issues: (1) the last hidden state

ht is used to predict the traffic flow at next time step only, so it is difficult to be applied to multi-step prediction; (2) traffic flow at different time steps may be of different importance for predicting results; (3) time information may significantly change traffic flow patterns, but it's hard to be fed into the network directly. Therefore, we developed an Attention-Based RNN architecture with Temporal Component (ABRNN_TC) for traffic flow prediction (shown in Figure 4.1). The proposed architecture includes three major components: encoder-decoder architecture, attention mechanism, and temporal component, to address the above issues, respectively.

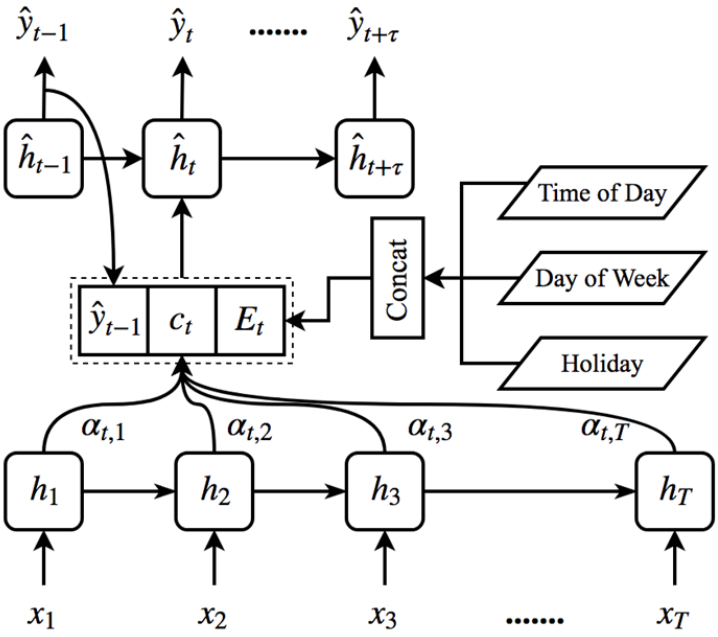


Figure 4.1: Attention-Based RNN architecture with Temporal Component

The encoder is a LSTM layer that reads an input sequence $x = (x_1, x_2, \ldots, x_T)$ and generates a dynamic context vector c from the input sequence for each time step in the decoder, where c_t is computed as a weighted sum of all hidden state of the encoder with Equation:

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j \tag{4.1}$$

the weight α_{tj} of each hidden state h_j is computed by Equation:

$$\begin{aligned} \alpha_{tj} &= \frac{e_{tj}}{\sum_{k=1}^{T} \exp(e_{tk})} \\ e_{tj} &= a\left(\hat{h}_{t-1}, h_j\right) \end{aligned} \tag{4.2}$$

where e_{tj} indicates how well the input at time step j and the output at time step t match, and a is modeled as a feedforward neural network which can be jointly trained with all the other components in the architecture.

Traffic flow has a strong correlation with the temporal factors, e.g., time of the day, day of the week and national holidays. The traffic pattern during weekdays, weekends and national holidays are usually very different. Previous work divided traffic flow into two different groups (e.g., weekdays and weekends) for prediction. In our model, a temporal component is added to handle these factors. As shown in Figure 4.1, we concatenate the temporal information (denoted as E_t), the time of the day (values from 0 to 95), the day of the week (from 0 to 6) and whether it is national holiday (0 or 1), to the context vector c_t.

Finally, another LSTM layer is used as decoder to generate the output sequence $\hat{y} = (\hat{y}_{T+1}, \hat{y}_{T+2}, \ldots, \hat{y}_{T+\tau})$. Unlike the LSTM described in Section 2.3.1, the decoder LSTM also includes the context vector c and the temporal component output E_t as input while updating the hidden state. Hence, the hidden state of the decoder $\hat{h}_t$ is calculated using Equation below.

$$\hat{h}_t = f\left(\hat{h}_{t-1}, \hat{y}_{t-1}, c_t, E_t\right) \tag{4.3}$$

where f represents a LSTM unit here.

4.1.2 Experiments and Evaluations

Dataset

The Caltrans Performance Measurement System (PeMS) [23] is a widely used dataset for traffic flow prediction [74, 85, 100, 195]. The traffic data was collected every 30 seconds from various types of vehicle detector stations throughout the state of California in the United States. We aggregated the sensory data into 15-min interval for each detector station. In this study, we used the dataset that was collected from 243 vehicle detector stations in district 5 (including Monterey, San Benito, etc.) from May 1, 2017 to Feb

28, 2018. The data of the first nine months was used for training, and the data of the remaining one month was used for testing.

Setup

For the input layer, we collected data from all 243 vehicle detector stations at previous r time steps, i.e., $x_{t-r}, x_{t-r+1}, \ldots, x_t$. The data includes both the relationship of 243 detector stations and temporal correlations. For the output layer, we predicted the traffic flow of the 243 stations at the next τ time steps, i.e. $x_{t+1}, x_{t+2}, \ldots, x_{t+\tau}$. The dimension of the input shape is $243r$; the dimension of the output shape is 243τ. As traffic flow volume of different stations may have different scales, the input data was further normalised to the range of [0,1].

We chose r from 2, 4, 8, 12, 24, 48, 96. After performing grid search, the best number of input time steps was 4, which means the traffic flow prediction mostly depends on the traffic flow of the previous one hour. The prediction performance dropped rapidly when r increased due to the difficulty in modelling long historical time sequences. We used the proposed method to predict traffic flow volumes in next 1 hour, 2 hours, and 3 hours, where τ is 4, 8, and 12 respectively. As the traffic flow of weekdays, weekends, and national holiday may have different patterns, we further collected time information as an additional input for the temporal component E_t in the proposed model.

With regard to the attention based RNN architecture, we need to determine the number of hidden layers, the number of hidden units, batch size, epochs, optimiser, and etc. Because training the attention based RNN model is time consuming, the number of hidden layers is set to 1 in this study. We chose the number of hidden units from 128, 256, 512, 1024 and the number of batch size from 64, 128, 256, 512. We used early stopping to avoid overfitting. After performing grid search, the best parameters were 512 for hidden units, 256 for batch size, 0 for dropout rate, 100 for epochs and Adam for optimiser. These experiments were run using Keras 2.1.5, Tensorflow 1.3, python 3.6, and Windows 10 on a laptop with a i7-6700HQ CPU, 8GB RAM and GTX-970M GPU.

Evaluation

Figure 4.2 presents the predicted results for three different vehicle detector stations at Freeway 1, Freeway 101 and Freeway 156. The actual traffic flow data is also plotted for comparison. The figure shows that the proposed model is able to learn the traffic flow

patterns and provides accurate traffic predictions. To evaluate the effectiveness of the proposed model, we used the root mean square error (RMSE) and mean absolute error (MAE) respectively.

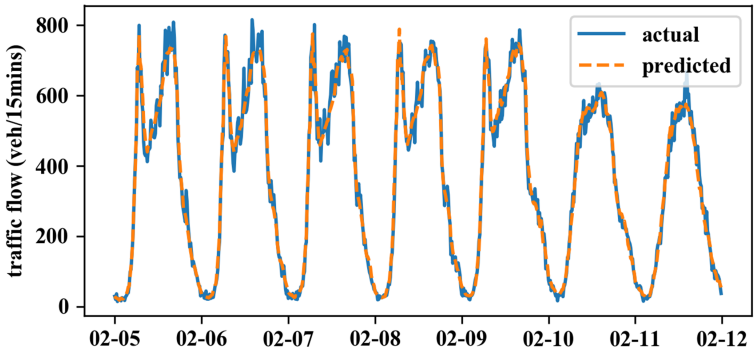
The performance of the proposed model was also compared with four other methods as briefly explained below.

- **AVG:** It is a simple method that calculates the average traffic flow of each station at specific time (e.g. 9 AM on Monday).
- **k-NN:** It finds the k most similar traffic flow patterns to the current traffic flow and predicts future time steps on the basis of the averaged future data of the k patterns.
- **Seq2Seq:** It uses a standard RNN with LSTM units to encoder the input sequence into a context vector and another RNN to make predictions iteratively.
- **Attention-Based Recurrnet Neural Network (ABRNN):** It introduces an attention mechanism to adaptively select the weight of hidden states from the encoder to produce the output sequence.

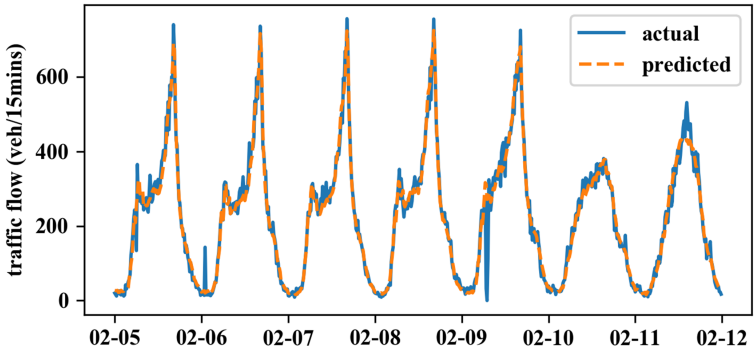
Table 4.1: Performance Comparison of Different Models for Traffic Flow Prediction

Method	1-hour prediction		2-hours prediction		3-hours prediction	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
AVG	6.86	4.42	6.86	4.42	6.86	4.42
k-NN	4.95	3.26	4.86	3.15	4.89	3.14
Seq2Seq	3.94	2.78	4.34	3.03	4.57	3.16
ABRNN	3.92	2.74	4.29	2.99	4.53	3.15
ABRNN+TC	**3.89**	**2.71**	**4.26**	**2.96**	**4.48**	**3.08**

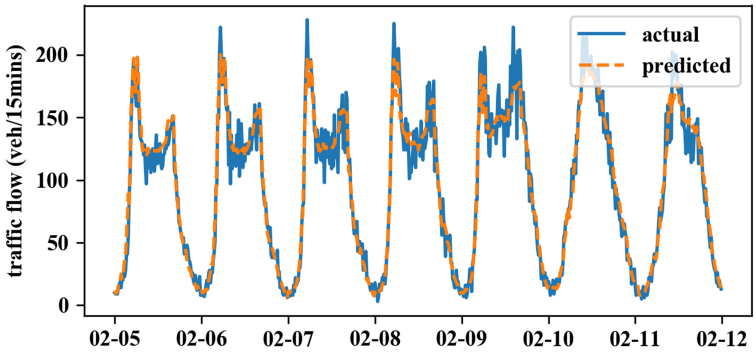
The best results of each method under different parameters were reported in Table 4.1. As expected, simply using the statistical AVG method led to inaccurate results. Following the idea of [27], using k-NN provided better and robust traffic prediction results. In general, the deep learning models, i.e., Seq2Seq and ABRNN, outperformed other methods, which has also confirmed by other previous studies [74, 85, 100, 195]. Compared to the standard Seq2Seq model, ABRNN models further improved the prediction accuracy due to the positive effects of the attention mechanism. Since there is not long temporal dependency in traffic flow data, only limited improvement could be observed with the addition of the



(a) Station at Freeway 1



(b) Station at Freeway 101



(c) Station at Freeway 156

Figure 4.2: Traffic flow prediction results for three stations with the ABRNN_TC model

attention mechanism. We expect attention-based mechanisms will be able to provide more notable improvement when applying to data with longer temporal correlations, e.g., air quality, water quality, etc. When adding the temporal information to the model, the MAE of attention based RNN model further decreased by around 2.2%. It should be noted that we only use the temporal information as the external data in this study. It is also possible to include more external factors that are relevant to traffic flow for further improvement, e.g., traffic speed, weather condition, accidents, and so on.

Use Case: Traffic Anomaly detection

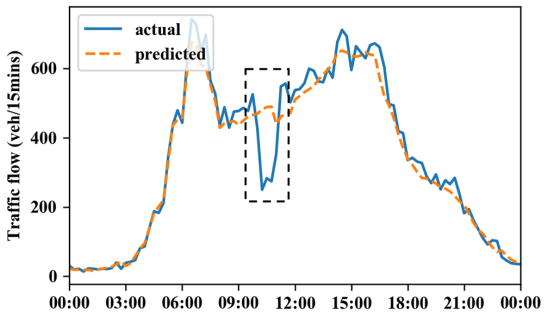
Traffic flow data follows a more or less recurring pattern. Meanwhile, it may vary abnormally due to various traffic events, road conditions, and other external factors. In these cases, the predicted traffic flow and the actual traffic flow may have significant differences. One interesting application is to analyse traffic anomalies to detect real-time traffic incidents or events. We demonstrate a use case that applies the proposed method together with social media information to detect traffic incidents and traffic events.

Since the proposed method is able to capture traffic flow patterns, an unusually high prediction error is likely to indicate a real-world traffic anomaly. For example, in Figure 4.3 it shows that there are three traffic anomalies on highway 101 in February 2018. Although a threshold can be set to extract the temporal and spatial information of traffic anomaly, details (i.e., event type, cause and impact) of the traffic event is still missing.

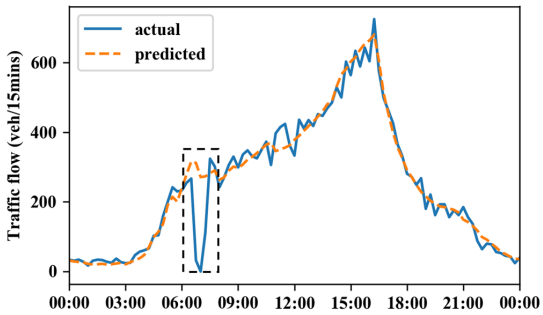
While social media data published by the citizen sensors differs semantically from sensing data generated by the physical devices to a great extent, it can be used as an important complementary source for traffic related applications. We further used the temporal and spatial information collected from the detected traffic anomalies to filter traffic related tweets posted by trusted organisations on Twitter. In Table 3, details of the traffic accidents, e.g., overturned truck or car crash, can be collected from tweets published by CaltransD5.

Table 4.2: Traffic event description collected from social meida platform

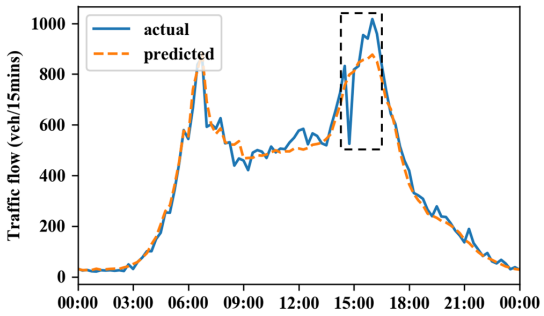
	Time	Location	Event
a	2018-02-02 9:00 - 12:00	Highway 101 off-ramp at Traffic Way	Overturned truck
b	2018-02-09 6:00 - 8:00	Highway 101 near Highway 58	Overturned semi-truck
c	2018-02-13 14:00 - 17:00	Highway 101 on Santa Maria Bridge	2-car crash



(a) Traffic event detected at 10AM on 2nd Feb.



(b) Traffic event detected at 7AM on 9th Feb.



(c) Traffic event detected at 3PM on 13th Feb.

Figure 4.3: Three Traffic events on Highway 101 detected from sensor data

4.2 Summary and Discussion

In this chapter, we select traffic flow prediction task as a typical example of deep learning from sensory data. Traffic flow prediction is an important while complex problem in transportation modeling and management. Many uncertain, non-linear and stochastic factors could have large influence on the prediction performance. With the recent development in deep learning, researchers have applied deep neural networks for the traffic flow prediction problem and achieved promising results. However, existing studies still have some issues unaddressed, e.g., the models only predict the traffic flow at next time step while travelers may need a sequence of predictions over the next few hours to make better, long-term decisions; external factors are (e.g., day of the week, national holiday) usually not well considered during prediction. To address these limitations, we propose an Attention Based

Recurrent Neural Network with Temporal Component (ABRNN_TC) model for multi-step traffic flow prediction. The experimental results show that with the addition of the attention mechanism and the temporal component, the deep model can capture traffic patterns accurately and produce superior prediction results over other baseline methods.

As the traffic flow data usually follows a recursive pattern (as shown in Figure 4.2), the attention mechanism can better extract historical traffic data of different importance for prediction. For other time series forecasting applications that do not have this historical dependency, the attention-based recurrent neural network may not outperform the traditional RNN encoder-decoder architecture. In addition, the proposed model does not consider the relationship between different sensors, and all sensor readings are represented as vector inputs for each time step. If this relationship is well modelled, for example, using graph neural networks, better prediction performance is expected.

Chapter 5

Learning from Multi-modal Data

You never change things by fighting the existing reality. To change something, build a new model that makes the existing model obsolete.

-Buckminster R. Fuller

Smart city is a typical Cyber-Physical-Social (CPS) system, which usually collects, processes and analyses data of different types and modalities. It is common that different sources may publish incomplete data in different modalities about the same phenomenon. Data from different sources should complement and knowledge discovered should reinforce each other, e.g. a traffic anomaly that is not inferred from traffic sensor observations might be clearly explained by a number of tweets. Nevertheless, the challenging problem is how to design a framework for processing such multi-modal data which differs greatly in the level of granularity and semantic meaning. Our work in this chapter aims to exploit data of different modalities while complementary to each other to extract trustworthy knowledge and improve classification performance.

In this chapter, we present two smart city applications with multi-modal data: Traffic Event Detection (Learning from sensor and textual data) in Section 5.1 and Crisis-related Data Classification (Learning from image and textual data) in Section 5.2. Besides showing multi-modal network that process data with multiple modalities, we also combine it with Generative Adversarial Network and Adversarial Training techniques to extend its abilities in semi-supervised learning and robust training respectively. Section 5.3 summarises the chapter and discusses some further issues related to our method.

5.1 Semi-supervised Traffic Event Detection

Intelligence Transportation Systems (ITS), as an instance of smart city applications, aim to discover knowledge from traffic related data collected from a city environment for the efficient management of transportation and mobility in a city. Lv et al. exploited historical traffic flow data for traffic prediction [100]; Song et al. engaged GPS records with millions of anonymous users for human mobility prediction [158]; Anantharam et al. collected social media data for traffic event detection [12]. However, most of these existing studies collect and analyse data from either the physical world [100, 158] or social world [12, 59], i.e. data of just one modality or from one source.

With the rapid development of ITS, the generated traffic data collected from loop sensors, GPS, cameras and social media, is exploding. Researchers believe that we have entered the era of big data transportation [100]. Much of the transportation research focus has shifted towards the processing of massive amounts of data continuously generated within a city environment. However, most of the existing studies only process data of single modality and require a large amount of labelled data, which is usually not practical in real-world, big data applications. This inspires us to design a semi-supervised learning framework for traffic event detection with the rich amount of unlabelled data and the extremely limited amount of labelled one.

Deep learning techniques are also considered as ideal candidates for knowledge fusion and integration, e.g., the research [189, 69] fuses representations learned from text, visual and audio. Generative Adversarial Network (GAN) [54] is one of the most influential models in recent deep learning research. The adversarial learning framework has been adopted in a number of tasks, such as learning representations for realistic image generation [54], novelty detection [144], and semi-supervised learning [160].

Figure 5.1 depicts the overview of the proposed multi-modal traffic event detection system. It is assumed that when a traffic event occurs, some kind of data characterising the event might be generated at different sources, e.g., pedestrians might post incident information on Twitter, or readings of the traffic sensor might show some different patterns. With the historical data, an event detection model (based on semi-supervised deep learning) can be effectively trained and used to detect future traffic events in real time. Consequently, detail about the event could be transferred to a traffic management centre (TMC) and disseminated to transportation users after verification.

The main contribution of this study is the design and evaluation of a multi-modal Gen-

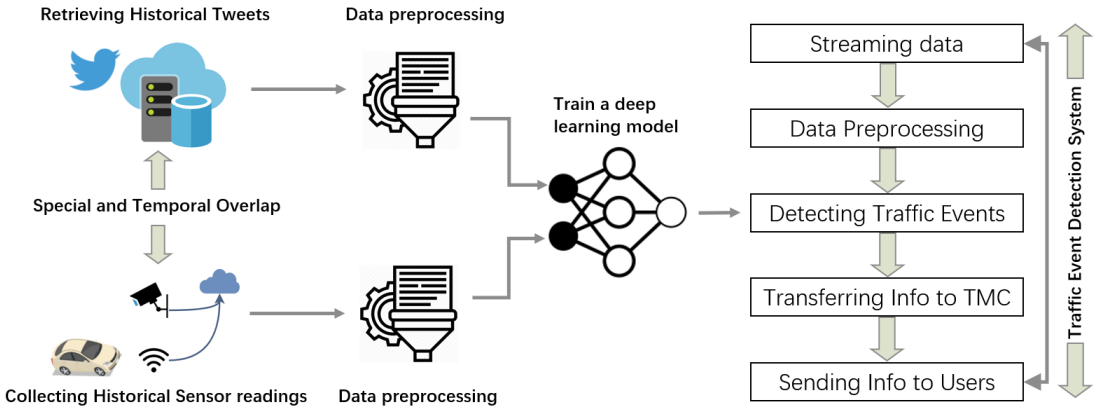
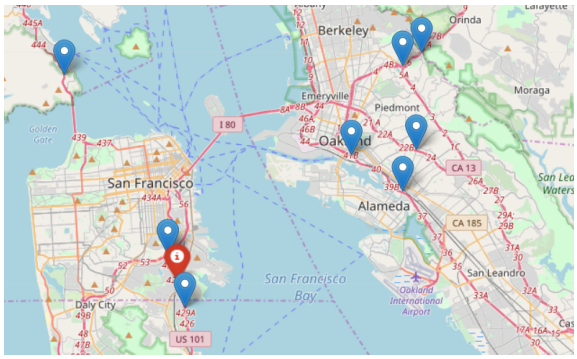


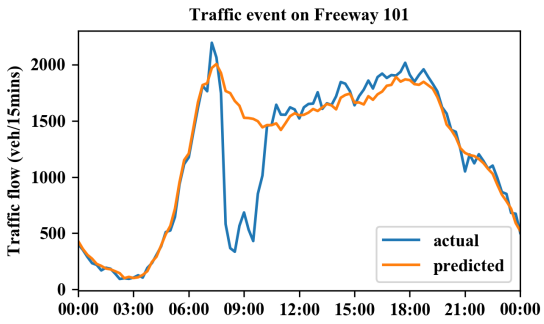
Figure 5.1: Overview of a traffic event detection system.

erative Adversarial Network (mmGAN) for the traffic event detection and classification. The proposed network attempts to address the two main limitations of existing studies on integrating data analysis of different modalities and extremely limited amount of labelled data in big data applications. A particularly novel aspect of the network is the employment of semi-supervised learning based on generative adversarial training. To our best knowledge, this is the first work to identify and classify traffic events with both sensor and social media data in a semi-supervised manner. The model has been evaluated on a large, real-world dataset, which contains 20 millions traffic flow readings and 8 millions tweets from the San Francisco Bay Area over a period of 4 months. The results confirmed that mmGAN can effectively learn useful representations characterising the multi-modal data simultaneously.

The rest of this part is organised as follows. In Section 5.1.1, we review some of the representative methods in processing and analysing sensor data and social media textual data in the intelligent transportation domain. In Section 5.1.2, we describe in detail the design of the semi-supervised, multi-modal Generative Adversarial Network for traffic event detection and classification, and the algorithm for semi-supervised training. In Section 5.1.3, we conduct a number of experiments with the proposed method as well as several baseline models on the same dataset, and discuss the evaluation results.



(a) Traffic events reported by authorities



(b) Traffic event detected from sensor observations

Tweet #1:
ID: 396296780930756609
Time: 2013/11/1 8:24:50
Location: [37.703, -122.394]
Text:
#Brisbane accident on US-101 NB at Harney Rd #BayArea #Traffic

Tweet #2:
ID: 396301955904966657
Time : 2013/11/1 8:45:23
Location: [37.713, -122.396]
Text:
In traffic and low on gas on highway 101

(c) Traffic event detected from social media data on Twitter

Figure 5.2: Illustration of the traffic events detected from cyber, physical and social worlds.

5.1.1 Related Work

Traffic events may be caused by many factors, e.g. accidents, traffic hazards, weather conditions, and traffic control. By analysing data collected from the cyber, physical and social worlds, traffic events can be detected and classified. Traffic events are normally reported by transportation authorities, with a possible delay in most of the cases. Figure 5.2a shows some events reported by the Department of Transportation on November 1st, 2013 (marked in blue and red). Figure 5.2b and Figure 5.2c show one event (the red one in Figure 5.2a) are signified by sensor data and social media data, respectively. Reporting these events could be much earlier if data is properly analysed. Besides traffic flow data and social media data, there are also many studies analysing data collected from GPS, smartphones, and cameras for traffic event detection. Based on data types, related research

can be categorised into sensor data based, social media data based, and multi-modal data based methods.

Methods using sensor data: With the rapid development in ITSs, the amount of traffic sensor data collected from GPS [208], loop sensors [100], smartphones [40] and cameras [207] is exploding and available to the public. Sensor observations usually follow a recurring pattern, but may vary abnormally due to traffic incidents, road conditions, social events, and other factors. For example, as illustrated in Figure 5.2b, the blue curve shows the actual traffic flow readings and the orange curve shows the predicted traffic flow series. A sudden drop of actual traffic flow at around 9 AM may represents a potential event and can be detected by anomaly detection approaches. Studies in [208] and [40] develop methods to detect sudden changes in GPS data collected from smartphones and taxi traces to identify incidents, traffic jams, and social events, and further discover when, where and how the event happened.

Methods using social media data: The large amount of social media data, covering nearly everything happening around the world, is easily accessible and has become valuable for research in data mining and knowledge discovery, e.g., sentiment analysis, event detection, and recommendation. In contrast to sensor observation data, social media data has some attractive features, e.g. it covers far more areas and topics, can be collected at low cost, and has high-level semantics understandable to human users. For traffic event detection purpose, millions of geotagged tweets can be acquired from Twitter in real-time and classified using various methods, for example, Conditional Random Fields [12], Latent Dirichlet Allocation [182], and deep neural networks [37].

Methods using multi-modal data: Studies [189, 69] have shown the use of deep learning models that fuses representations learned from text, visual and audio, and show superior performance than single modality models. However, the use of multi-modal data in smart city applications or more specifically traffic event detection task is still very limited. City data from different sources and worlds may has different characteristics. For example, the trustworthiness of the data collected from the social world may be questionable because of social spams. Data from the physical world usually has low-level semantics and may not be always available due to sensor faults or communication failure; furthermore, coverage of the sensor deployment may be limited. In many situations, data from the two worlds can be complementary, and by analysing them together, it is argued that more comprehensive and trustworthy knowledge can potentially be discovered. Previous approaches, i.e., [124] and [13] exploit sensor data for traffic anomalies detection, then search social media data

with the detected time and location, and further describe or explain the anomalies with the social media textual data. However, these studies do not consider and process the data of different modalities simultaneously. In a sense, they have not fully exploited the potential of the complementary data. Their limitations are similar to those that process and analyse data of individual modalities. In this part, we address this issue by designing a multi-modal feature learning component which process both sensor and social media data simultaneously.

5.1.2 multi-modal Generative Adversarial Network

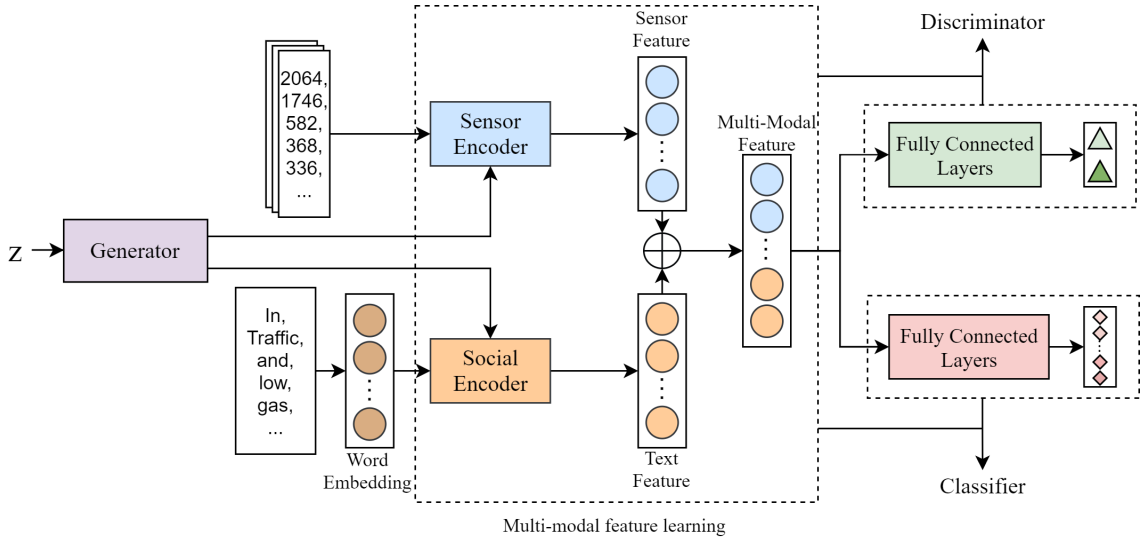


Figure 5.3: Multi-modal feature learning from both sensor time series and text embeddings

We designed a multi-modal Generative Adversarial Network (mmGAN) for Semi-supervised, multi-modal traffic event detection. The overall architecture of the proposed mmGAN is shown in Figure 5.3. In this architecture, the multi-modal feature learning component is used to encode the input data into numerical vectors and transform the data of different modalities into representations which can be simultaneously processed by one network. The output from each encoders is concatenated to form a multi-modal feature representation for the generative adversarial learning. The semi-supervised Generative Adversarial learning process takes as input the data of multiple modalities and attempts to not only discriminate if the the data is real or generated, but also classify it. It aims to exploit

the complementary sensor and social media data for better traffic event detection and classification, with limited amount of labelled data and large amount of unlabelled one.

Multi-modal Feature Learning

Our current study considers traffic related data of two different modalities, i.e., sensor data which is usually represented as time series, and social media tweets which are represented as short texts. The multi-modal feature learning architecture transforms different data into a unified multi-modal feature representation as shown in Figure 5.3. There are two types of encoders: the Sensor Data Encoder component is for sensor input processing (shown in Figure 5.4) and Social Data Encoder for social media text (shown in Figure 5.5). The two deep network components extract features from the sensor time-series and twitter messages, respectively. The extracted features are concatenated to form one multi-modal feature representation, which is used in the multi-modal Generative Adversarial Network for detecting and classifying traffic events.

Sensor Data Encoder

To extract features from sensor time-series, we use the Recurrent Neural Network (RNN) as the core module. We select the LSTM unit in this study which can solve the exploding and vanishing gradient problems of vanilla RNNs. The detailed LSTM equations have been shown in Equation 2.4 and the process of using RNN for traffic flow data has been discussed in Section 2.3.1.

In Figure 5.4, two RNN layers are used in the Sensor Data Encoder to extract representations. As traffic sensor observation may vary abnormally due to traffic events, the first RNN layer is pre-trained and aims to predict traffic flow sequences given historical observations. Potential traffic events are represented by the difference between the actual sensor reading and predicted values (referred to as residuals). The calculated residual values during traffic event usually should be much larger than the one at normal period, as shown in Figure 5.2b. The residual values are the input to the second RNN layer, which aims to extract the representation for the potential events.

Social Data Encoder

The Social Data Encoder component attempts to extract an effective representation for the short social media messages. The input to the Encoder is a sequence of words in a

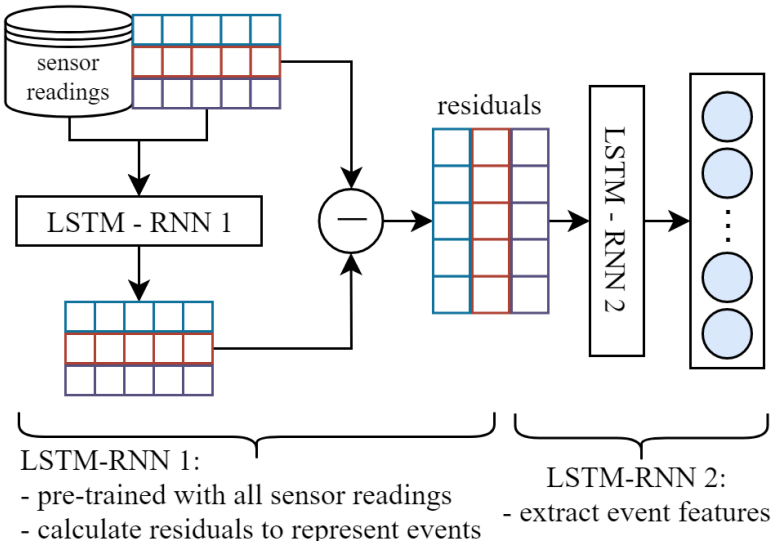


Figure 5.4: Sensor Encoder Architecture

tweet, each of which is represented as a word embedding vector. They are initialised with the word embedding pre-trained on 400 million twitter posts [52]. A tweet with n words can be represented as $S_{1:n} = (S_1^d, S_2^d, ..., S_n^d)$, where d is the dimension of the embedding vector.

The architecture of encoder is shown in Figure 5.5. The way that it extracts textual representations from tweets is similar to the one proposed in [80]. It consists of a convolutional layer and a max pooling layer. In the convolutional layer, a convolution filter has a size of $h \times d$, where h is the window size and d is the width of filter equal to the word-vector dimension. Sliding the filter across the matrix $S_{1:n}$ produces a feature map s^j with size $(n-h+1)$ which is represented as $s^j = [s_1, s_2, ...s_i, ..., s_{n-h+1}]$, where s_i is calculated with Equation 5.1.

$$s_i = ReLU(W_c \cdot S_{i:i+h-1}) \tag{5.1}$$

where $ReLU$ is an activation function, W_c represents the weights of filter, $S_{i:i+h-1}$ represents the contiguous h word embedding vectors and $(\cdot)$ is the dot product between weights W_c and word vectors $S_{i:i+h-1}$. As shown in Figure 5.5, the coloured dashed lines in the convolutional layer represents this convolutional learning process, where different colours denote different filters.

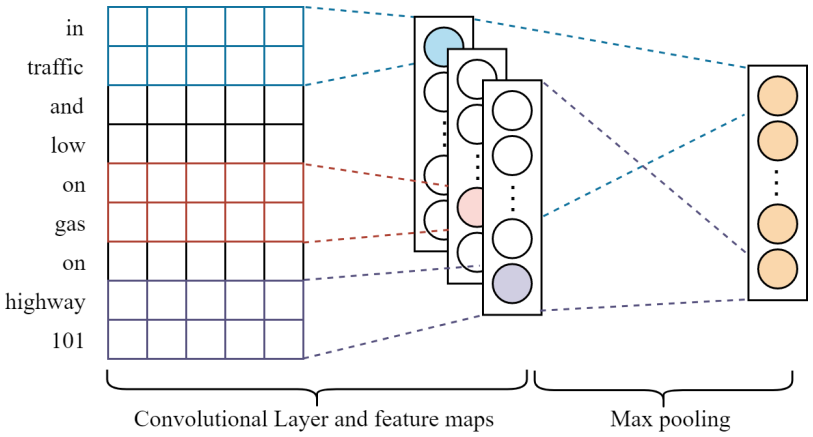


Figure 5.5: Social Data Encoder architecture

With K different filters, K feature maps $s = [s^1, s^2, ..., s^K]$ are generated. We apply a max-pooling operation to each feature map with Equation 5.2.

$$m = [\delta(s^1), \delta(s^2), ..., \delta(s^K)] \tag{5.2}$$

where $\delta(s^j)$ denotes the max-pooling process, which selects the maximum value from the feature map s^j. Intuitively, the convolution operation extracts local features into higher level representations in the feature maps, and the max-pooling extracts the distinguishing aspects of each feature map while reducing the output dimension. The final social data representation m with size K is then concatenated with sensor data representation for further multi-modal classification.

semi-supervised Generative Adversarial learning

Due to the fact that only limited amount of labelled data is available in any big data applications, it is not appropriate to use standard deep learning methods for supervised tasks. We noticed in our experiments that results generated using either the Sensor Data Encoder, Social Data Encoder or Multi-modal feature component is unsatisfactory. To this end, we extend the model to a semi-supervised architecture based on the Generative Adversarial Network (GAN) [54]. A standard supervised model, e.g., classifiers, can be extended to a semi-supervised one by adding samples from the generator G in GAN to the dataset and labelling them with a new class, "generated", denoted as $y = K + 1$ [160,

146, 38, 212]. Inspired by the idea, we design a multi-modal, semi-supervised adversarial training architecture for traffic event detection and classification.

As shown in Figure 5.3, the architecture consists of three components: a generator G, a discriminator D, and a classifier C. G and D in mmGAN are trained with conflicting objectives. G takes in a noise vector z and produces traffic data of two modalities: sensor observations and social media texts (both in the form of numerical vectors). D takes in multi-modal feature vector and predicts if it is a sample from the real data or G. G is trained to maximise the probability that D makes a mistake, while D is trained to minimise the probability that it makes a wrong prediction. Through this adversarial training process, features that could distinguish real samples from the generated ones are learned in an unsupervised way. Thus, the multi-modal feature component could capture useful representations from the large amount of unlabelled data through this adversarial training process.

C is a standard multi-class classifier that also takes in multi-modal feature vector and attempts to predict a correct label for an input. As the multi-modal feature learning component is shared by both D and C, the three components can be jointly optimised. By utilising the large amount of unlabelled data (with G and D) and a limited amount of labelled date (with C), the proposed model forms a semi-supervised architecture, and result in performance improvement.

The loss function L for training the generator L_g, discriminator L_d and classifier L_c are shown in Equation 5.3, Equation 5.4, and Equation 5.5 respectively.

$$L_g = -\mathbb{E}_z \log D(G(z)) \tag{5.3}$$

$$\begin{aligned} L_d = &- \mathbb{E}_{x_s, x_t \sim p_{data}(x_s, x_t)} \log D(x_s, x_t) \\ &- \mathbb{E}_z \log(1 - D(G(z))) \end{aligned} \tag{5.4}$$

$$L_c = -\mathbb{E}_{x_s, x_t, y \sim p_{data}(x_s, x_t, y)} \log C(x_s, x_t) \tag{5.5}$$

where x_s and x_t represents sensor data input and twitter word embedding input, respectively. The detailed training process of the proposed mmGAN is summarised in algorithm 1.

Algorithm 1 mmGAN Training Algorithm

Input: unlabelled multi-modal input (x_s, x_t), and labelled multi-modal input (x_s, x_t, y);

for the number of training iterations **do**
  Draw m noise samples;
  Draw m samples from unlabelled multi-modal input (x_s, x_t);
  Perform gradient descent on the parameters of D according to Eq. 5.4 on the combined mini-batch of size $2m$;
  Draw m noise samples;
  Perform gradient descent on the parameters of G according to Eq. 5.3;
  Draw m samples from labelled multi-modal input (x_s, x_t, y);
  Perform gradient descent on the parameters of C according to Eq. 5.5;
end for

5.1.3 Experiments and Evaluation

We have conducted extensive experiments using a large, real world, multi-modal dataset, which was prepared by interlinking two large datasets that have been widely used in existing research and a specific dataset constructed by ourselves. Performance of the proposed method was evaluated and compared to a number of state-of-the-art supervised learning methods.

Dataset

The Caltrans Performance Measurement System (PeMS) [23] provides large amount of traffic sensor data that has been widely used by the research communities. The data is collected every 30 seconds from many vehicle detector stations that report at 5-minute interval throughout the state of California in the United States. We used the traffic flow data in the San Francisco Bay Area from August 2013 to November 2013 and further aggregated the readings at 15-minute interval. The resulting dataset contains around 20 million traffic flow readings from 1,649 traffic detector stations and data size is around 13 GB.

We reused the geo-tagged twitter dataset published in [12], in which more than 8 million tweets were collected from August 2013 to November 2013 for the San Francisco Bay Area. The size of the twitter dataset is around 2 GB. Most of the tweets were posted by ordinary people that cover a variety of topics. We also collected geo-tagged tweets that report traffic event information from official accounts, i.e., @TotalTrafficSF, from August 2013 to

November 2013 with the twitter API.

To create the multi-modal traffic dataset, we first filtered traffic-related tweets with a list of keywords, e.g. "traffic, block, delay, highway, freeway, accident, incident, construction". We matched the tweets with the sensor data that has temporal (two hours) and spatial (one kilometer) overlapping, therefore creating 2,227 pairs of multi-modality samples. We used the (California Highway Patrol) CHP incident dataset collected from the PeMS [23] to assist the labelling process. The CHP incident dataset contains the detailed time, location, duration and incident types (e.g., traffic hazard, traffic collision, etc.), which we used to label the multi-modal instances based on spatial and temporal overlaps. The remaining instances that are not covered in the incident dataset are manually labelled. Our task is to consider both the sensor and tweet data simultaneously and categorise it into one of the two classes: (1) **Traffic event**, representing a non-recurring event that generates an abnormal change in traffic and transportation capacity. The examples of non-recurring events include traffic crashes, disabled vehicles, road construction, vehicle fire, etc. The current work is to inform users and agencies the occurrence of an ongoing traffic event if there is any. We will consider the case of multi-class classification in the future work, which would provide users more intuitive information with specific event types; and (2) **Traffic information (non-traffic event)**, reporting daily traffic conditions, past traffic events, new traffic rules, traffic advisory, and any other information on transport infrastructures. The number of traffic event and the number of traffic information for both training and test sets are reported in Table 5.1.

Table 5.1: Distributions of the Multi-modal Datasets

	Training Dataset	Test Dataset	Total
Traffic Event	1390	155	1545
Traffic Info	614	68	682
total	2004	223	2227

Setup

Each input sample to the model consists of a sensor observation sequence and a twitter message (in the form of a sequence of word embeddings). As there may be multiple sensors reporting the same event, we built an input block for sensor data with a block size of 10.

The dimension of the sensor input shape is 16×10, where 16 is the number of time steps in 4 hours. For social media text input, we represented each word with a word embedding of 400 dimension. Most of the tweets contain less than 15 words, so we only considered the first 15 words in each tweet. The dimension of a twitter input is 15×400.

The model is trained with 90% of the data and tested with the rest 10% of the data. We performed a grid search to determine the best parameters for the proposed mmGAN: in the Social Data Encoder, the window size of filter was set to 2, and the dimension of the hidden units in both Social Data Encoder and Sensor Data Encoder was set to 32. For the two fully connected layers in discriminator D and classifier C, the hidden size was set to 32. The number of batch size was 64; the dropout rate was set to 0.5; the Adam optimiser with early stopping was used to avoid overfitting.

Baseline Models

Data of single modality (i.e., either sensor data or social media data) can also be used to discover traffic events. We re-implemented and tested the following two baseline models with only data of single modality: (1) **Support Vector Machine (SVM)** is a popular kernel method for supervised learning tasks and has been widely used to process sensor data and social media data, e.g., time series prediction [175] and twitter classification [129]. We implemented two SVM models using the sensor time series and social text embeddings, respectively, to detect traffic events; and (2) **Sensor Data Encoder (SeE)** and **Social Data Encoder (SoE)** have been explained in Section 5.1.2, and shown in Figure 5.4 and Figure 5.5, respectively. They were used separately to extract features from sensor data and social media data. To detect and classify traffic events, a fully connected layer with hidden size of 32 and a sigmoid output layer were added at the top of the two models.

We also re-implemented and tested the following three baseline models with data multiple modalities for performance comparison: (1) **Multi-Modal Neural Network (MMN)** was implemented as a classifier and its architecture is similar to the component c in the mmGAN architecture; (2) **RNN** has been primarily used to to extract features from data with strong temporal characteristics, e.g., sensor data [170] and social media data [37]. In our implementation, after using RNN to process sensor data and social media data separately, the extracted feature vectors were concatenated and fed into a fully connected layer for prediction; and (3) **CNN** has also been successfully used to extract features from sensor data [184] and social media data [37]. We used CNN to extract fea-

tures from sensor and social media data separately, and concatenated the feature vectors for prediction.

Evaluation

We used the standard evaluation metrics for assessing classification performance, i.e., accuracy, weighted average recall, precision, $F1$ and AUC score. We report the mean results on models trained with 10-fold cross-validation. The evaluation results of the proposed mmGAN and the baseline models are shown in Table 5.2.

Table 5.2: Performance Comparison of Different Models for Traffic Event Detection

Data	Sensor Data Only		Social Data Only		Multi-modal Data		
Models	SVM	SeE	SVM	SoE	RNN	CNN	mmGAN
Accuracy	60.83	68.23	76.10	82.24	84.17	83.45	**87.17**
Precision	59.53	67.93	76.76	83.19	84.30	83.21	**87.40**
Recall	60.83	68.28	76.10	82.24	84.17	83.34	**87.17**
F1	59.87	64.36	76.30	82.37	84.05	83.21	**87.16**
AUC	60.06	64.94	82.31	90.87	91.43	91.03	**93.44**

The most notable observation is that the overall performance of those models that process and analyse multi-modal data simultaneously is better than those with only data of a single modality in all metrics. This observation obviously confirmed our expectation that exploiting complementary data of different modalities does enable us to extract trustworthy knowledge and improve classification performance. Among the three models that process multi-modal data, mmGAN outperformed both CNN and RNN with 3% improvement in accuracy, 3.1% in precision, 3% in recall, 3.11% in $F1$ and 2.01% in AUC. This showed the effectiveness of the multi-modal learning and the adversarial training: the sensor and social data encoder learn representations from both types of data; the generator and the discriminator compete with each other and improve each others' performance at the same time. As a consequence, the classifier making use of the shared component in the discriminator is able to improve its own performance with the multi-modal data, even though its differs greatly in the level of granularities and semantic meaning.

Table 5.2 shows that models (i.e., SVM, Social Data Encoder, and Sensor Data Encoder) which process and analyse single modality data can also detect and classify events successfully with certain degree. In general, as sensor data contains much noise and many

missing values, e.g., no nearby sensors to report such traffic event, models using the sensor data produced the worst performance among all the methods. On the contrary, social media data contains more obvious features, e.g., traffic events related keywords, which helped extract more informative representations and generate better results compared with processing sensor data. With single modality data, the proposed Sensor Data Encoder and Social Data Encoder outperformed the SVM in terms of all metrics, which showed that the two models could extract reasonably good representations from the sensor and text data, respectively.

To evaluate the performance of the mmGAN with a limited amount of labelled data, we compared mmGAN with the MMN. MMN is a classifier and its architecture is identical to the component c in the mmGAN architecture. Their performance in terms of accuracy with different amount of labelled data is plotted in Figure 5.6. It can be seen that the proposed mmGAN outperformed MMN when we shrank the size of the training data. When the size of the labelled data was very small, e.g., $5\% - 30\%$ of the whole labelled data, mmGAN notably outperformed MMN. This confirmed that the discriminator d could learn better representations from both large amount of unlabelled and very limited amount of labelled data through the generative adversarial training. Ultimately, it contributed to improve the performance of classifier c. It can also be seen that when more and more amount of labelled data was used, their performance tended to converge. As the number of traffic related tweets that can be matched with sensor readings is limited, we select a portion of the paired data as labelled data and treat the rest part as unlabelled data. We will consider using the additional unlabelled data (13GB sensor data and 2GB tweet data) of each modality separately in an unsupervised manner to further improve the results in the future work.

Case study

We provided 4 examples extracted from the dataset in Figure 5.7 and 5.8. They were misclassified by models using a single modality data, but correctly classified with mmGAN using multi-modal data. We show the sensor readings and social media content in four subfigures. In each subfigure, the coloured lines show the patterns of real sensor readings, where x-axis represents time of day and y-axis denotes traffic flow in 15 minutes. The colours of lines represent different sensors close to (within 1 kilometer) a particular event, and the number of sensors reported in each event could be different. The corresponding

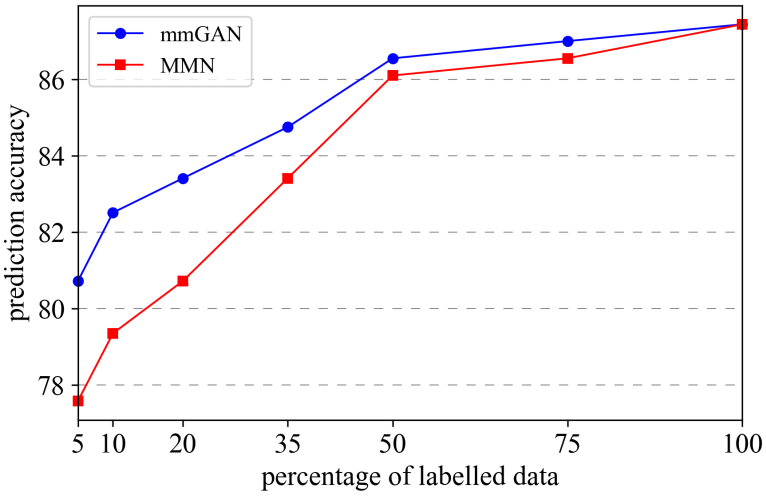
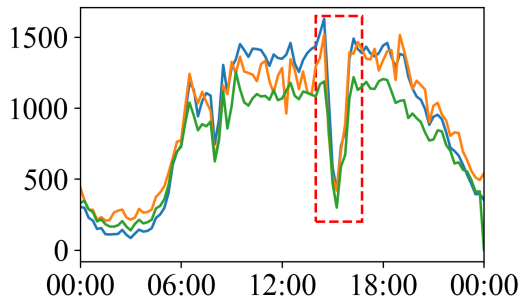
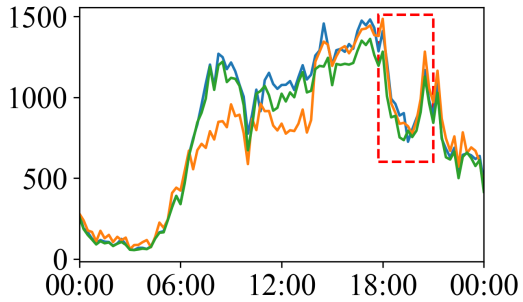


Figure 5.6: Classification accuracy of mmGAN and MMN with different amount of labelled data.

tweet messages are shown below.



(a) 2013-09-17 15:01:42
Forgot how much I hate traffic on the 101. And why didn't I get a tesla rental at the Avis counter?

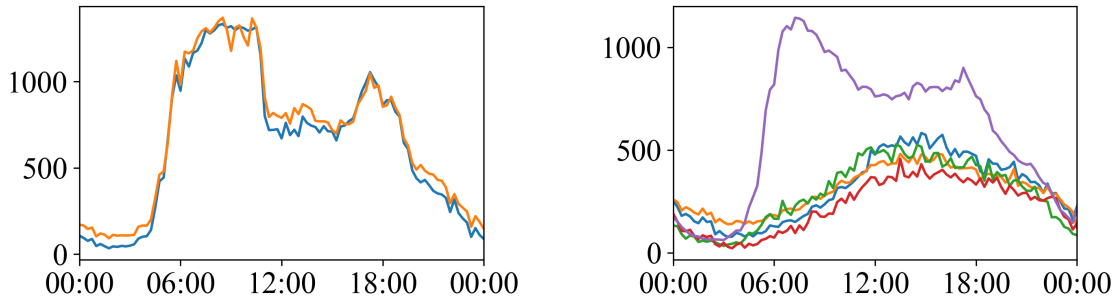


(b) 2013-10-02 19:13:00
Sun is going down. Stuck in traffic. Ugh. Looks like we're going to be late to #maroon5.

Figure 5.7: Two traffic events correctly detected by mmGAN but misclassified by others. Diagrams show the patterns in the real sensor data and captions represent the corresponding tweet messages.

Figure 5.7 shows two traffic events that were successfully detected by mmGAN but were missed by others. The tweet messages did not provide enough evidence to identify

whether it was a traffic event, so the Social Encoder model wrongly classified the two cases as non traffic events. By adding the sensor data into the proposed mmGAN, the cases were correctly classified as traffic events.



(a) 2013-09-03 19:08:49
Reporter on the San Jose news just said higher employment = more traffic. Now we can get stuck in traffic.

(b) 2013-09-02 08:29:35
SJPD seek driver after double-fatal accident.

Figure 5.8: Traffic related information detected by mmGAN but misclassified by the single modality model

Figure 5.8 shows two examples misclassified as real-time traffic events by models using only the social media data, but successfully spotted by mmGAN. As the two tweets contain traffic event related keywords, e.g. stuck, traffic, and accident, the Social Encoder model categorised them as traffic events. However, the traffic flow data at the same location during the same time period showed normal traffic patterns. By using both type of data, the cases were correctly classified as non traffic events by mmGAN.

5.2 Adversarial Learning from Crisis-related Data

At times of natural disasters, social media platforms such as Twitter and Facebook are considered vital information sources that contain a variety of useful information such as reports of injured people, infrastructure damage, missing people, etc. [70]. Processing social media data to extract life-saving information which is helpful for organizations in preparedness, response, and recovery of an emergency [6]. Studies [116, 115] build Deep Neural Networks for information classification and knowledge discovery. However, DNN models are vulnerable to adversarial examples. Small perturbations to the input will cause

the DNN models to generate incorrect results [168]. To improve the robustness of DNN models, adversarial examples and adversarial training techniques are widely investigated and have become one of the most influential trends in recent deep learning research.

In this part, we propose a Multi-modal Adversarial Training (MMAT) method for the crisis-related tweets classification. Different from most of the past studies that focused on textual content only, the proposed method is tailored to learn useful representations from both image and text data simultaneously. In particular, adversarial training is applied to improve the robustness of the neural network against adversarial examples. To our best knowledge, this is the first work to classify multi-modal crisis-related social media data with adversarial training techniques. Experimental results demonstrate the proposed MMAT method is able to attain significant improvement of prediction accuracy.

In Section 5.2.1, we review some of the representative work in crisis-related data classification and adversarial training. In Section 5.2.2, we describe in detail the design of the multi-modal network and its adversarial training process. In Section 5.2.3, we conduct a number of experiments with the proposed model and compared with baseline models on the four datasets.

5.2.1 Related Work

In this section, we present the related work from two perspectives. The first part demonstrates the representative applications of crisis-related data classification on social media. The second part presents some recent studies about adversarial examples and adversarial training for image and text.

Crisis-related tweets classification: During natural disasters, users may post text and image data on social media platforms to report information about infrastructure damage, injured people, cautions and warnings. For example, Figure 5.9 shows two crisis-related tweets during California wildfires and Mexico earthquake. Effectively processing twitter messages in real-time could help city organizations gain situational awareness and provide better response actions. Analysis of big crisis data on social media with machine learning techniques is increasingly studied, e.g., the work in [75] used conditional random fields to extract valuable information from twitter messages; the work in [76] published a large word2vec embeddings trained with 52 million crisis-related tweets and classify crisis tweets with Support Vector Machines (SVM), Naive Bayes (NB), and Random Forest (RF) methods. With the advances in deep learning, much recent tweet classification work

is founded on DNNs; reported results significantly outperform conventional machine learning methods such as SVM, NB and RF. For example, the work in [116] and [115] presented a Convolutional Neural Network (CNN) based framework to capture salient n-gram information using convolution and pooling operations, so as to identify useful or crisis-related tweets. The work in [114] used pre-trained CNN to process images posted on social media during natural disasters to access the level of damage caused by disasters.

(a) Raining Ash and No Rest: Firefighters Struggle to Contain California Wildfires.

(b) Earthquake leaves hundreds dead, crews combing through rubble in Mexico.

Figure 5.9: Illustration of crisis-related data collected from twitter.

Adversarial Training: Deep Neural Networks (DNNs) have achieved remarkable success in various tasks, e.g., image classification, natural language processing, and speech recognition. However, DNN models are vulnerable to adversarial examples. Small perturbations that are imperceptible to humans will cause the DNN models to generate incorrect results when adding into the input. Such issue poses security concerns regarding the uses of DNN models in security-sensitive applications. To improve the robustness of DNN models, adversarial attack and defence techniques are widely explored recently and have become an influential direction for deep learning research. Various defensive techniques against adversarial examples for deep neural networks in the image domain have been proposed [168, 55]. Recent studies [126, 147] also shown the potential of adversarial attacks and defence in the texts domain. After crafting adversarial examples, adversarial training process that injects such examples into training data could help increase DNN robustness [168]. In our study, we explore the adversarial training technique for multi-modal data and apply it to the robust classification of crisis-related tweets.

5.2.2 Multi-modal Adversarial Training

We first present the architecture for Multi-Modal neural Network (MMN), and its visual-CNN and text-CNN components that extract information from multiple modalities into a multi-modal representation for tweet classification. Then, we explain the adversarial training process, which retrains the MMN with crafted adversarial examples for images and text for more robust crisis-related tweets classification.

Multi-modal neural network

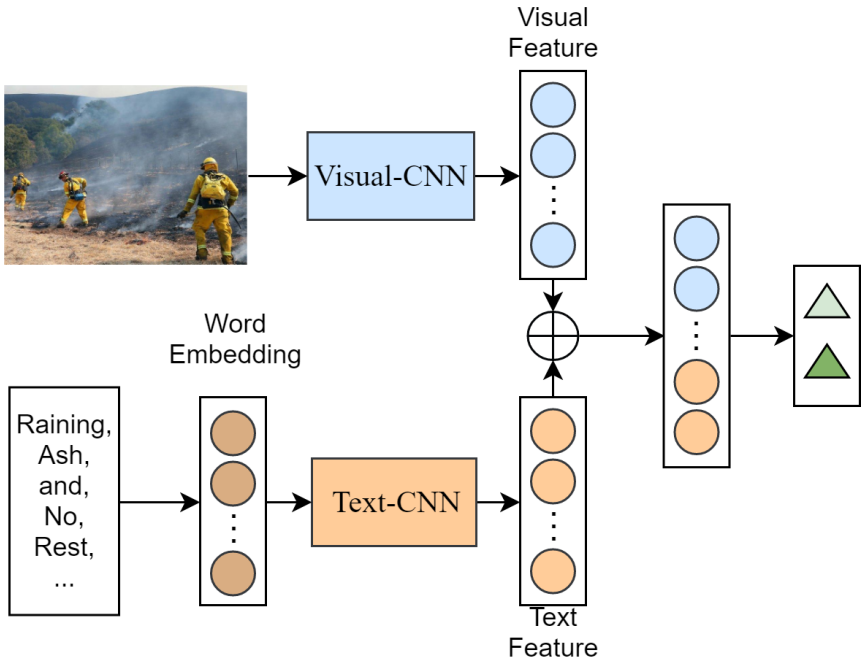


Figure 5.10: Multi-modal neural network with both image and text data as input

Majority of past studies only focused on textual content for crisis-related data classification. In this study, we consider data of two different modalities: image and text. Image data is usually represented as a 2 or 3-dimensional matrix with real values, while textual tweets are represented as word sequences. The multi-modal neural network aims to learn feature from both types of data and fuse into a unified representation as shown in Figure 5.10, where the Visual Convolutional Neural Network (Visual-CNN) is for image input processing and Text Convolutional Neural Network (Text-CNN) is for text input

precessing (shown in Figure 5.11).

The Visual Convolutional Neural Network (Visual-CNN) component aims to extract visual features from images of the tweets. The Visual-CNN consists of two sets of convolutional and max pooling layers, followed by a flatten layer and a fully-connected layer. It should be noted that the performance of visual feature extractor may further be improved with higher quality image input and pre-trained CNN models, e.g., VGG [153], ResNet [64] and DenseNet [73].

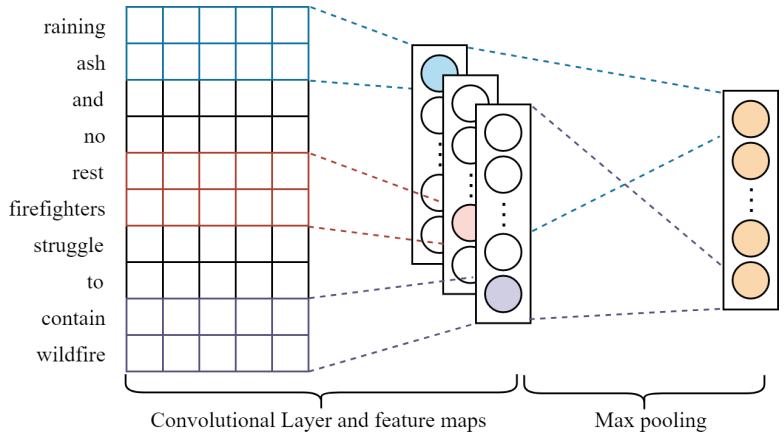


Figure 5.11: Text Convolutional Neural Network (Text-CNN) architecture

The Text Convolutional Neural Network (Text-CNN) component attempts to extract an effective representation for the short social media texts. The input to the Text-CNN is a sequence of words, each of which is represented as a word vector. They are initialised with the glove vectors pre-trained on 2 billion twitter posts [127]. The architecture of Text-CNN component is shown in Figure 5.11, which is identical to 5.5 that has been presented in Section 5.1.2. The extracted visual and text representations from Visual-CNN and Text-CNN are then concatenated into a multi-modal feature representation, which is further used for detecting and classifying crisis-related tweets.

Adversarial Training

To improve the robustness of deep neural network against adversarial examples, adversarial training techniques are widely investigated in recent years. By adding small perturbations η to original input x, the adversarial example x' aims to fool the neural network f so that $f(x') \neq y$. Such small perturbations are imperceptible to humans eye but may cause

deep learning models' misclassification. A robust deep neural network model should not change its output for these small perturbations in its input. Therefore, the adversarial training process utilise such adversarial examples as an augmented dataset to increase model robustness [168]. Fast Gradient Sign Method (FGSM) [55], shown in Equation 5.6, is one of the most popular methods used to calculate perturbations and generate adversarial examples.

$$\eta = \epsilon sign(\nabla J(\theta, x, y)) \tag{5.6}$$

where θ is the parameters of a model, x and y are the input and associated labels, $J(\theta, x, y)$ is the cost of the model. For the multi-modal twitter dataset used in this study, x includes both image input x_i and text input x_t, and thus the associated perturbations are η_i and η_t.

Because the input space of image data is continuous, image adversarial examples can be directly created with $x_i' = x_i + \eta_i$. While the input space of texts is discrete, a perturbed word vector may not represents any word. We cannot directly set the word vector to specific real values in the word embedding space. Therefore, FGSM could not be directly applied to texts to generate adversarial examples. Inspired by work in [126], we craft adversarial tweets by replacing original words with new words that have the largest projection length in the direction of perturbation η_t. To keep semantic similarity, adversarial words are selected from a candidate set C, where only n most similar words in the word embedding space are included for each word in a tweet. The process of word replacement is shown in Equation 5.7.

$$w' = \underset{w' \in C}{\arg\max}(w' - w) \cdot \eta_t \tag{5.7}$$

where w' is the adversarial word for w, C is the candidate set, and η_t represents the perturbation for word w calculated by Equation 5.6. Iteratively, replacement for each word in a sentence can be determined and finally form an adversarial example for tweet that may be misclassified by deep learning models. In addition, as we may not want all word to be replaced in a tweet, a threshold for $(w' - w) \cdot \eta_t$ can be set so as to update only the most impactful words for the model's prediction results.

After generating adversarial examples for image and text, the MMN can be retrained with the augmented dataset for better robustness against adversarial examples. The whole

multi-modal adversarial training process is shown in Algorithm 2.

Algorithm 2 Multi-modal Adversarial Training Algorithm

1: Input: labelled multi-modal input (x_i, x_t, y); MMN classifier f
2: Train MMN classifier f with (x_i, x_t, y)
3: Obtain perturbation (η_i, η_t) for (x_i, x_t) with Equation 5.6.
4: Create Image adversarial x_i' with $x_i' = x_i + \eta_i$
5: Create Text adversarial x_t' by iteratively update words with Equation 5.7.
6: Data augmentation with:
$(x_i^{aug}, x_t^{aug}, y^{aug}) = data\_aug((x_i, x_t, y), (x_i', x_t', y))$
7: retrain MMN classifier f with augmented dataset $(x_i^{aug}, x_t^{aug}, y^{aug})$

5.2.3 Experiments and Evaluation

Dataset

We used the multi-modal crisis-related twitter dataset published in [6], where 16,097 tweets were collected during seven disasters, specifically California wildfires, Hurricane Harvey, Hurricane Irma, Hurricane Maria, Iraq-Iran earthquake, Mexico earthquake, and Sri Lanka floods. We select four datasets for performance evaluation, which are California wildfires, Hurricane Harvey, Mexico earthquake and Sri Lanka floods.

Our goal is to categorise images and text data simultaneously into one of the two classes: (1) **Informative** represents the tweet or image is useful for humanitarian aid. The examples of informative include cautions and advice, infrastructure and utility damage, injured or dead people, affected individuals, missing or found people, etc. The objective is to providing assistance to people who need help in order to save lives, reduce suffering, and rebuild affected communities. (2) **Not informative** represents the tweet or image is not useful for humanitarian aid, e.g., images showing banners, logos, and cartoons. In the original dataset, image and text are labelled as informative and not informative separately for each tweet. We combine them together so as to extract all informative tweets that have either an informative image or informative text.

Setup

Each input sample to the model consists of an image and a textual message (in the form of a sequence of word vectors). As the dimensions of each image could be different, we

resized each image into 64×64 and preprocessed values into range 0 to 1. For textual messages, we represented each word with a glove word vector of 100 dimensions. Most twitter messages are short in length, so we only considered the first 15 words in each tweet and result in 15×100 dimensions.

We performed a grid search to determine the best parameters for the proposed MMAT. In the Text-CNN, the window size of filter was set to 2, and the dimension of the hidden units in both Visual-CNN and Text-CNN was set to 64. For generating adversarial examples, θ in FGSM is set to 0.1 and the threshold for crafting adversarial text is set to 0.5. The number of batch size was 32; the dropout rate was set to 0.5; the Adam optimiser with early stopping was used to avoid overfitting. These experiments were run using Keras 2.1.5, Tensorflow 1.3, python 3.6, and Windows 10 on a laptop with a i7-6700HQ CPU, 8GB RAM and GTX-970M GPU.

Baselines

We implemented some baseline models that process either data of single modality or data of multiple modalities for performance comparison.
Baseline Models with Data of Single Modality:

- **Visual-CNN** and **Text-CNN** are explained in Section 5.2.2, and shown in Figure 5.10 and Figure 5.11. CNN is used separately to extract features from image data and text data as described in Section 5.2.2. A fully connected layer with the hidden size of 64 and a sigmoid output layer were added at the top of these two models to generate the final prediction.

- **Visual-CNN**$_{adv}$ and **Text-CNN**$_{adv}$ are the adversarial training versions of Visual-CNN and Text-CNN. After training Visual-CNN and Text-CNN with the original dataset, adversarial examples for image and text are crafted separately with the equations in 5.6 and 5.7. Such adversarial examples are then added into training data to retrain Visual-CNN and Text-CNN and named Visual-CNN$_{adv}$ and Text-CNN$_{adv}$.

Baseline Models with Data of Multiple Modalities:

- **Multi-modal neural network (MMN)** uses CNN to extract visual and textual features from image and text data separately, and concatenated the feature vectors

for prediction. Detailed architecture of MMN is explained in 5.2.2, and shown in Figure 5.10. The architecture of MMN has been employed in various different types of supervised applications, e.g., audio-visual speech enhancement [69], fake news detection [189] and Image-sentence ranking [83]. In this study, we used the MMN method as a baseline to evaluate the performance of adversarial training technique for the multi-modal dataset.

Evaluation

We assess classification performance with four twitter datasets of different natural disasters, i.e., California wildfire, Harvey Hurricane, Mexico Earthquake and Srilanka floods. The evaluation results of the proposed MMAT and the baseline models are shown in Table 5.3.

Table 5.3: Performance Comparison of Different Models for Crisis-related Data Classification

Methods	Visual		Text		Multi-modal	
	CNN	CNN_{adv}	CNN	CNN_{adv}	MMN	MMAT
California Wildfire						
original	86.16	85.53	86.79	87.42	**88.68**	88.05
adversarial	45.91	82.39	76.73	84.28	67.30	**88.05**
Harvey Hurricane						
original	80.67	80.22	88.09	88.09	**89.21**	87.42
adversarial	35.51	78.43	73.26	**86.07**	73.26	84.04
Mexico Earthquake						
original	72.66	75.54	82.73	84.89	84.89	**86.33**
adversarial	30.94	71.94	71.94	78.42	58.71	**79.86**
Srilanka Floods						
original	67.96	66.02	92.23	92.23	**93.20**	**93.20**
adversarial	36.89	66.02	80.85	**89.32**	79.61	**89.32**

Table 5.3 shows that models (i.e. Visual-CNN, and Text-CNN) which process images and text separately can detect and classify tweets successfully with a certain degree. In general, as image data usually contain much redundant information and sometimes not related to the topic, the Visual-CNN model produced the lowest accuracy among all the

methods in these four datasets. On the contrary, text data contains more obvious features, e.g. crisis-related keywords, which help extract more informative representations and result in better accuracy. One notable observation from the table is that the overall performance of multi-modal models, i.e., MMN and MMAT, that process multi-modal data simultaneously is better than the models with data of single modality. This observation shows that exploiting complementary data of multiple modalities helps extract more comprehensive knowledge and improve classification performance.

Another notable observation from the table is that the adversarial training technique can significantly improve models' classification accuracy against adversarial examples range from 6.48% to 42.92%. For text data, the adversarial examples cannot be directly crafted with perturbations calculated from FGSM. Thus, the improvement of text adversarial training is lower than the improvement of image adversarial training. As the crafted adversarial examples don't contain additional information, the adversarial models MMAT may not outperform the baseline MMN in the original datasets (e.g. California wildfire dataset and Harvey Hurricane dataset). Among the three adversarial models that retrained with adversarial examples, the proposed MMAT outperforms Visual-CNN$_{adv}$ and Text-CNN$_{adv}$ with the highest accuracy in California wildfire dataset 88.05%, Mexico earthquake dataset 79.86%, Srilanka floods dataset 89.32%, and with slightly lower accuracy in Harvey Hurricane dataset 84.04%. This confirms the effectiveness of multi-modal adversarial training that the crafted adversarial examples for image and text could be used simultaneously to help improve the robustness of DNN models with multiple modalities.

Case Study

Two tweets extracted from the California wildfire dataset and their adversarial examples are shown as below. The MMN model is vulnerable to these multi-modal adversarial examples, and generated incorrect prediction results.

Figure 5.12a shows a tweet that describes the situation of California wildfire and thus is labelled as informative. In Figure 5.13a, although the tweet has the keyword "California wildfire", it's related to fake news and should be classified as not informative. The MMN could correct classify these two tweets as informative and not informative with 89.1% and 97.2% confidence separately. Adversarial examples of these two tweets, as shown in Figure 5.12b and Figure 5.13b, are generated with the equations in 5.6 and 5.7. Although the modified images have some visible perturbation noise and the crafted sentence includes

(a) ICYMI: Why California Wildfires Are Infernos In October.

(b) ICYMI: Why California Wildfires Are Infernos Was October.

Figure 5.12: adversarial example of an informative tweet

some word replacements that may not follow grammar rules, they still have the same semantic meaning and should be classified as the same. However, the MMN misclassified them as not informative and informative with 63.5% and 90.1% confidence separately. These two examples illustrate multi-modal networks are vulnerable to adversarial examples, and thus an adversarial training process is necessary for improving the model's robustness for these life-saving tasks.

5.3 Summary and Discussion

Modern smart city applications usually process and analyse big city data from different sources, often in multiple modalities. While the data might be noisy, incomplete or inconsistent, it is complementary to each other and potentially enables more valuable knowledge to be extracted. In this chapter, we present two smart city applications with multi-modal data: Traffic Event Detection (Learning from sensor and textual data) in Section 5.1 and Crisis-related Data Classification (Learning from image and textual data) in Section 5.2.

Semi-supervised Traffic Event Detection: Traffic event detection is an important

(a) Fake News: NO Illegal Muslim From Iran Arrested For Starting California Wildfire.

(b) Fake Reports: NO Criminals Christians Of Iraq Arrested For Starting California Wildfire.

Figure 5.13: adversarial example of a not informative tweet

and complex task in smart transportation modelling and management. We address this problem using semi-supervised deep learning with data of different modalities, e.g., physical sensor observations and social media data. Unlike most existing studies focusing on data of single modality, the proposed method makes use of data of multiple modalities that appear to complement and reinforce each other. In addition, it is also impractical to obtain large amount of labelled data in real-world applications. We propose the multi-modal Generative Adversarial Network, a semi-supervised, deep learning based model that can process data of multiple modalities in a unified framework, for traffic event detection and classification. The evaluation results clearly showed the advantages of the mmGAN over other models with or without multi-modal data in terms of precision, accuracy and $F1$ in classification. Furthermore, the generative adversarial training process with large amount of unlabelled and limited amount of labelled data could indeed help extract more useful knowledge than other baseline models. In the future, we plan to further refine the proposed model so that it can process more types of data for other smart city applications, e.g., GPS traces, image and wearable sensor data. As data from different sources cannot always be matched, an interesting direction is to extend the proposed model to be compatible with both single

modality input and multi-modality input. An important problem being considered is to apply the attention mechanisms to model the temporal interplay of multi-modal data and learn better representations from it, rather than simply concatenating representations learned from data of individual modality. Another future work is to extend the current model to support multi-class or even multi-label classification, which would provide users more intuitive knowledge.

Adversarial Learning from Crisis-related Data: During natural disasters, users may post text and image data on social media platforms to report information about infrastructure damage, injured people, cautions and warnings. Timely and effective processing and analysing tweets can help city organisations gain situational awareness of the affected citizens and take timely operations. With the advances in deep learning techniques, recent studies have significantly improved the performance in classifying crisisrelated tweets. However, deep learning models are vulnerable to adversarial examples, which may be imperceptible to the human, but can lead to model's misclassification. To process multi-modal data as well as improve the robustness of deep learning models, we propose a multi-modal adversarial training method for crisis-related tweets classification on social media. To extract valuable knowledge from tweets, a multi-modal network is applied to learn features from both images and text that can complement each other. As the nature of social media, image and text data posted by ordinary people are usually noisy, inconsistent and may not follow grammar rules. We proposed a multi-modal adversarial training framework to improve the robustness of crisis-related tweets classification. The evaluation results clearly showed the advantages of the proposed MMAT over other models with data of single and multiple modalities. In the current work, we focused on extracting informative tweets from not informative ones. We plan to extend the current model to support multi-class or multi-label classification, which would provide users with more detailed information, e.g., infrastructure damage, injured people, affected individuals, etc. For textual adversarial examples, generating high-quality sentences that follow grammar rules could be another future research direction. Moreover, we plan to further refine the proposed model and extend to other smart city applications, e.g., transportation, energy, and environmental protection.

Chapter 6

Learning From Social Media Data

Research is what I'm doing when I don't know what I'm doing.

-Wernher von Braun

'human as sensors' or 'citizen sensing' [151] has become a popular phenomenon for which humans are not only the data users, but also the data providers. It allows the general public to collect, analyze, report and disseminate information, enabling them to better perceive and understand the world. Meanwhile, it is crucial for the development of social IoT [122], an integral part of the Cyber-Physical-Social systems (CPSS) [202, 185]. Enormous amount of social media data can be collected and further processed and analyzed in various downstream tasks which may have great influence on human society. For example, users may post real-time traffic information on Twitter, which facilitates traffic event detection [37]. Other examples include reports of injured or missing people, infrastructure damage, and warnings and cautions; which all help crisis/disaster assessment and emergency response [115, 76].

To extract useful information and knowledge from social media data, Natural Language Processing (NLP) techniques are usually adopted. Under the traditional supervised learning paradigm, DNNs have become unbeatable in terms of classification performance, provided that there are sufficiently large amounts of well labeled examples. However, they usually break down when there is not sufficient labeled data, which is common in real world applications. The ability to transfer the knowledge gained while solving one problem and applying it to a different but related problem (referred to as transfer learning) can alleviate

this issue. One notable example of using transfer learning in NLP, so far, is to pre-train representations on a large unlabelled text corpus and then adapt the trained representation to a supervised target task. A number of pre-trained models have been developed very recently, e.g. word2vec [103], GloVe [127] and Bidirectional Encoder Representations from Transformers (BERT) [42], that have been applied to various tasks, e.g. image caption generation [179], sentiment analysis from social media [204], and text classification in smart city application [115]. Besides the use of pre-trained models, the research community has also shown great interest in other forms of transfer learning, e.g. domain adaptation [49], multi-task learning [43], zero-shot learning [78], etc.

In this chapter, we focus on fine-grained text classification task with limited amount of labelled data from social media. We present two smart city applications (Crisis-related data classification and COVID-19 tweet classification) in Section 6.1 and Section 6.2 respectively. We combine the state-of-the-art NLP model (BERT) with adversarial domain adaptation and zero-shot learning techniques to learn from unlabelled data. We summarise the chapter and discusses some future directions in Section 6.3.

6.1 Domain Adaptation for Crisis-related Data

Twitter. Around 200 billion tweets are posted on Twitter per year, which covers nearly everything happening around the world. Such large amount of data is easily accessible and has become valuable resources for various research in data mining and knowledge discovery, e.g., sentiment analysis [150], traffic event detection [37], and crisis-related data classification [116, 115]. During crisis events, people may post useful information such as reports of injured or missing people, infrastructure damage, warnings and cautions via social media platforms [70]. Processing social media data to extract life-saving information is helpful for organizations to gain situational awareness, learn about urgent needs and decide on response actions [116]. Many machine learning techniques, i.e., conditional random fields [75], support vector machines [76] and Deep Neural Networks, [115] could be used to extract valuable information, identify useful tweets or assess damage from crisis-related tweets.

Recently, Deep learning has shown great performance in crisis-related data classification [116, 115]. However, training a good deep learning model usually requires a large amount of labeled data which is not easy to collect especially in the early stage of a crisis. To address this issue, a deep neural network model should learn representations from a labeled source

dataset and adapt to a related unlabeled target dataset by dealing with the shift in data distributions between the source and the target domains. This process is referred to as domain adaptation.

In this part, we propose a BERT-based Adversarial Domain Adaptation method (BERT-ADA) for crisis-related tweet classification. Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained language model that achieved state-of-the-art results in various text classification tasks. We apply adversarial domain adaptation to further improve the performance of the model on unlabeled datasets. To our best knowledge, this is the first work that combines BERT with adversarial domain adaptation for crisis-related tweet classification. Experimental results demonstrate the proposed BERT-ADA model is able to gain significant improvement over other baseline models.

We review some related work of adversarial domain adaptation in Section 6.1.1. In Section 6.1.2, we describe details of the BERT-ADA architecture, the pre-trained BERT model, adversarial domain adaptation technique and the model's training process. Section 6.1.3 presents the results of a number of experiments directed at the proposed model and compared with baseline models.

6.1.1 Related Work

As the recent studies of crisis-related tweets classification have been reviewed in Section 5.2.1. We show some representative domain adaptation studies in this section.

Adversarial domain adaptation: The traditional supervised learning paradigm has become very good at predicting an accurate output with good training models, but usually break down when we do not have sufficient labeled data. The ability to transfer the knowledge gained while solving tasks in the source domain and applying it to a different but related target domain (referred to as domain adaptation) can be applied to alleviate this issue. Adversarial domain adaptation [49] aims to obtain a representation where the source domain and target domain are close to each other while keeping good performances on the source labeling task. Recently, it has been applied to natural language processing tasks, e.g. sentiment analysis [51], fake news detection [189], and crisis-related tweets classification [5], and achieved state-of-the-art performance. In this paper, we adopt the idea of adversarial domain adaptation and combine it with the state-of-the-art NLP model (i.e. BERT [42]) for classifying crisis-related tweets in an unsupervised manner.

6.1.2 BERT-based Adversarial Domain Adaptation

We demonstrate our BERT-ADA model on a tweet classification task to support crisis response efforts. Let (X_s, Y) be the set of labeled crisis tweets for a source domain and X_t be the set of unlabeled tweets for a target domain, where $Y \in \{1, 2, ..., K\}$ is the class labels for tweets X_s. The goal is to train a domain-invariant model with parameters θ that can classify any tweet in the target event without having any information about its label in Y. The overall architecture of BERT-ADA, shown in Figure 6.1, contains three components: BERT-based feature extractor f, label classifier C and domain classifier D. Details of each component and the training process are explained separately in this section.

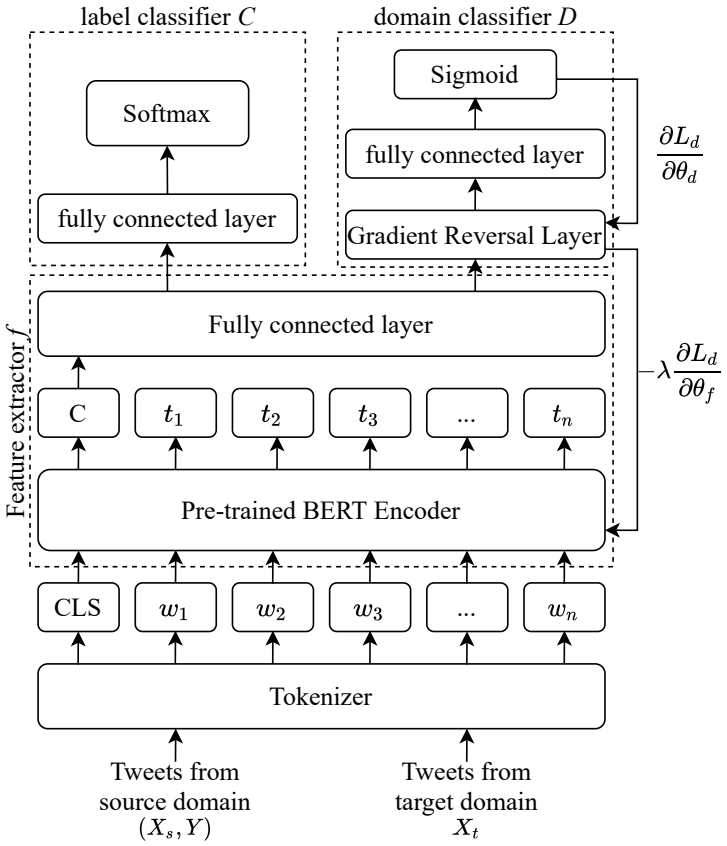


Figure 6.1: BERT-based Adversarial Domain Adaptation (BERT-ADA) architecture

Bidirectional Encoder Representations from Transformers (BERT)

We follow the standard BERT model as presented in Section 2.3.4 for tweet classification. In this work, we fine-tuned the pre-trained BERT model on crisis-related tweet classification tasks and achieve significant accuracy improvement over other commonly used deep learning models. Two fully connected layers and a softmax classifier are added to the top of the pre-trained BERT model to predict the probability of labels. The whole classification process is trained using the standard supervised loss L_c as below:

$$\begin{aligned} &L_c(\theta_f, \theta_c) = \\ &- \mathbb{E}_{(x_s,y_s)\sim(X_s,Y_s)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log C(f(x_s, \theta_f), \theta_c) \end{aligned} \tag{6.1}$$

where $\mathbb{1}$ is an indicator function, f is the BERT-based feature extractor, C is the label classifier, while θ_f and θ_c denote the corresponding parameters of f and C.

Adversarial domain adaptation

The network described so far can learn features with the pre-trained BERT model and classify each tweet's type in the source domain (Affected individual, Infrastructure damage, donations, etc.). However, we aim to learn domain invariant features from different crisis events (e.g., Australia bushfire vs. Queensland flood) and generate accurate predictions for tweets in the target domain. This unsupervised domain adaptation task is achieved by adding a domain classifier D on the top of BERT-based feature extractor f with a Gradient Reversal Layer (GRL) in it. During the forward propagation, GRL acts as an identity function. While in the backpropagation process, GRL takes the gradient from its subsequent layer, i.e., $\frac{\partial L_d}{\partial \theta_d}$, multiplies it by $-\lambda$ and passes it through feature extractor f, i.e., $-\lambda\frac{\partial L_d}{\partial \theta_f}$. GRL ensures that the feature distributions of the target dataset are aligned to the distributions of the source dataset, which is not distinguishable by the domain classifier, and resulting in domain-invariant features. The domain classifier D which predicts whether a tweet is drawn from the source or the target domain is optimized with the loss function L_d:

$$\begin{aligned} L_d(\theta_f, \theta_d) = &- \mathbb{E}_{x_s \sim X_s} \log D(f(x_s, \theta_f), \theta_d) \\ &- \mathbb{E}_{x_t \sim X_t} \log(1 - D(f(x_t, \theta_f), \theta_d)) \end{aligned} \tag{6.2}$$

The feature extractor f and label classifier C optimize the following loss:

$$L(\theta_f, \theta_c, \theta_d) = L_c(\theta_f, \theta_c) - \lambda L_d(\theta_f, \theta_d) \tag{6.3}$$

During domain adaptation, feature extractor f is trained to maximize the probability that domain classifier D makes a mistake. Through this adversarial process, domain-invariant feature representations are learned in an unsupervised way. Based on this idea, we are seeking parameters $\theta_f, \theta_c, \theta_d$ that:

$$(\hat{\theta}_f, \hat{\theta}_c) = \arg\min_{\theta_f, \theta_c} L(\theta_f, \theta_c, \hat{\theta}_d) \tag{6.4}$$

$$\hat{\theta}_d = \arg\max_{\theta_d} L(\hat{\theta}_f, \hat{\theta}_c, \theta_d) \tag{6.5}$$

The feature extractor parameters θ_f minimize the label classifier loss through standard supervised training, while maximizing the domain classification loss through adversarial training, the parameters θ_c of the label classifier c minimize the label prediction loss, the parameters θ_d of the domain classifier d maximise its domain classification loss (since the gradient is reversed when back-propagate to the feature extractor).

Model training

The detailed training process is illustrated in Algorithm 3. The parameters of the BERT component is firstly initialized with the pre-trained BERT-Base model. Other parameters are initialized with random numbers sampled from a uniform distribution. In each iteration, two types of gradient are calculated based on Eq. 6.1 and 6.2 to update parameters $(\theta_f, \theta_c, \theta_d)$ in three model components. Finally, after the model converges, the distribution of data in the source domain and the target domain will become similar, thus the label of tweets from the target domain could be directly predicted via feature extractor and label classifier components.

Algorithm 3 BERT-ADA Training Process

Input: labeled source data (X_s, Y_s), and unlabeled target data (X_t);
Output: learned parameters $(\theta_f, \theta_c, \theta_d)$
initialize model parameters $(\theta_f, \theta_c, \theta_d)$;
for the number of training iterations **do**
    Draw m samples from labeled source dataset (X_s, Y_s);
    Perform gradient descent on the parameters θ_f and θ_c of according to Eq. 6.1;
    Draw m samples from labeled source dataset X_s;
    Draw m samples from unlabeled target dataset X_t;
    Perform gradient descent on the parameters θ_f and θ_d according to Eq. 6.2 on the combined mini-batch of size $2m$;
end for

6.1.3 Experiments and Evaluation

We have conducted extensive experiments using a crisis dataset, named CrisisLexT26. Performance of the proposed method was evaluated and compared to several state-of-the-art learning methods.

Dataset

CrisisLexT26 [119] is a crisis-related tweet dataset that contains 250 thousand tweets posted during 26 crisis events in 2012 and 2013. In each event, around 1,000 tweets were labeled by crowdsource workers according to informativeness, information types, and information sources. Our goal is to process tweet messages and categories each tweet's information type into one of the seven classes:

1. **Affected individuals**: deaths, injuries, missing, found, or displaced people.
2. **Infrastructure damage**: damaged buildings, roads, utilities and etc.
3. **Donations**: needs, requests, or offers of money, shelter, supplies, etc.
4. **Caution and advice**: warnings, tips and guidance.
5. **Sympathy and support**: thoughts, prayers, gratitude, sadness, etc.
6. **Other information**: useful information not covered by the above categories.
7. **not related**

We selected three crisis events from CrisisLexT26 for performance evaluation, which are Australia bushfire, Queensland floods, and West Texas explosion. After removing duplicate tweets, tweets less than three words and tweets labeled as not applicable, we got three crisis datasets contains 922, 929, and 747 labeled tweets respectively. The detailed distribution of three datasets is shown in Table 6.1.

Table 6.1: Distributions of three crisis datasets

Datasets	Australia bushfire (A)	Queensland floods (Q)	West Texas explosion (W)
Affected individuals	64	111	174
Infrastructure damage	65	110	47
Donations	37	54	39
Caution and advice	186	202	14
Sympathy and support	133	78	242
Other Information	305	225	180
not related	132	149	51
total	922	929	747

Setup

We used the pre-train BERT-Base model for extracting representations from tweets, which contains 12 Transformer blocks and 768 dimensions of the hidden unit. We preprocessed the tweet text to lower case and removed any punctuation, question marks or URLs before feeding into the model. As most of the tweets are short, we set the maximum number of tokens to 30. We performed a grid search to determine the best parameters for BERT-ADA. The dimension of the hidden units in fully connected layers was set to 64. During adversarial training, λ was set to 0.2. The batch size was 32. A dropout layer was added at the top of the pre-trained BERT encoder with its rate set to 0.5. The main challenge during model training is to train a stable model that balances the adversarial components, i.e. f and D, of the network. The reason they are difficult to train is that both the feature extractor component f and the domain classifier component D are trained simultaneously in the model. Every time the parameters of one component are updated, the nature of the optimization problem being solved is changed and thus fails to converge. In our experiments, we found that the domain classifier is weaker than the rest of the network.

To balance the adversarial components, we used the Adam version of stochastic gradient descent to train the model with a learning rate of 1e-7. For fine-tuning the supervised components f and C, we used the Adam optimizer with a learning rate of 1e-4. In addition, early stopping strategy was used to avoid overfitting. These experiments were run using Keras 2.1.5, Tensorflow 1.3, python 3.6, and Windows 10 on a desktop with an i7-9700F CPU, 32GB RAM and RTX-2070S GPU.

Baseline Models

We implemented four baseline models that are commonly used in NLP classification tasks for comparison.

- **Long Short-Term Memory (LSTM)**: A LSTM recurrent neural network was applied to classify tweets represented as word embeddings, which is introduced in Section 2.3.1.
- **Bidirectional Long Short-Term Memory (Bi-LSTM) [149]**: A bidirectional LSTM was used to incorporate both left and right context, which is also introduced in Section 2.3.1.
- **Convolutional Neural Network (CNN)**: A CNN to extract features from word embeddings with convolutional filters, max pooling layers and fully connected layers, which is identical to the Text-CNN introduced in Section 5.2.2.
- **Bidirectional Encoder Representations from Transformers (BERT) [42]**: To classify tweets by fine-tuning the pre-trained BERT model, which is introduced in Section 2.3.4.

The input to the LSTM, Bi-LSTM and CNN should be firstly represented as a sequence of word vectors before entering the model. The input word vectors were initialized with the GloVe vector pre-trained on 2 billion twitter posts [127] and kept fixed during the training process. We represented each word with a glove word vector of 50 dimensions. Most twitter messages are short in length, so we considered at most 30 words in each tweet, which resulted in 30×50 dimensions. The best dimension for the hidden units for LSTM, Bi-LSTM and CNN models was 128. The window size of filters for CNN was fixed to 2.

Adversarial Domain Adaptation Models

To illustrate the performance of the adversarial domain adaptation technique, we extend the four baseline models to Adversarial Domain Adaptation (ADA) versions, namely LSTM-ADA, Bi-LSTM-ADA, CNN-ADA and the proposed BERT-ADA. To achieve this, the domain classifier described in section 6.1.2 was added at the top of feature extractors in the LSTM, Bi-LSTM, CNN and BERT models. We use the same setup for domain classifiers in all four models.

Evaluation

We collected three twitter datasets corresponding to the three selected natural disasters, i.e., Australia bushfire (A), Queensland floods (Q), and West Texas explosion (W), and built six test cases for access domain adaptation performance: A→Q, Q→A, A→W, W→A, Q→W and W→Q. In each case, the dataset on the left side of the arrow is the source dataset and the dataset on the right side is the target dataset. For transfer baselines models, we directly use the labeled source dataset to train LSTM, Bi-LSTM, CNN and BERT models and test them with an unlabeled target dataset. For adversarial domain adaptation methods, we use both labeled source data and unlabeled target data to train LSTM-ADA, Bi-LSTM-ADA, CNN-ADA, and BERT-ADA models and test them by classifying target data. The evaluation results of the proposed BERT-ADA and other models are shown in Table 6.2.

Table 6.2 shows that the three transfer baseline models (i.e. LSTM, Bi-LSTM, CNN) can detect and classify tweets successfully within a certain degree of effectiveness. As the glove input vectors to these three models are the same, the prediction accuracies of these three models are similar, e.g., LSTM is slightly better for the A→Q and A→W tasks, while CNN is slightly better for the remaining tasks. When we compare these context-free models with the BERT model, we observe significant improvements in prediction accuracy for all six cases ranging from 0.83% to 6.04%. This observation demonstrates the effectiveness of fine-tuning a large pre-trained contextual model for crisis-related tweet classification.

Another notable observation from the table is that the adversarial domain adaptation technique can significantly improve all models' classification accuracy across all six test cases ranging from 0.46% (BERT to BERT-ADA in W→A) to 7.1% (Bi-LSTM to Bi-LSTM-ADA in A→W). This confirms the effectiveness of adversarial domain adaptation in learning domain-invariant features and improving classification performance for

Table 6.2: Performance Comparison of Different Models for Crisis-related Data Classification

Methods	Source → Target					
	A → Q	Q → A	A → W	W → A	Q → W	W → Q
Transfer Baselines						
LSTM	46.93	47.40	50.47	43.28	52.48	32.19
Bi-LSTM	46.72	48.37	49.26	43.06	51.94	32.83
CNN	46.82	49.24	49.40	46.31	53.95	34.66
BERT	52.97	53.55	52.54	48.49	54.78	35.62
Adversarial Domain Adaptation						
LSTM-ADA	47.90	49.13	54.75	45.44	54.08	34.98
Bi-LSTM-ADA	48.65	51.63	56.36	46.96	54.21	35.41
CNN-ADA	48.76	51.95	53.15	47.40	55.56	37.24
BERT-ADA	**55.34**	**57.64**	**58.20**	**48.95**	**56.91**	**38.81**

an unlabeled target dataset. Among all eight models, the proposed BERT-ADA models achieved the best accuracy performance in all six cases. From these improvements, we can conclude that the BERT model, along with domain adaptation, is an effective method to leverage labeled data in the source domain and unlabeled data in the target domain for tweet classification.

6.2 Zero-shot Text classification for COVID-19 Tweet

COVID-19 pandemic has become a hot topic on Twitter. In this part, we explore the automatic classification of COVID-19 related tweets on social media without any labelled data for training using the zero-shot learning technique. Zero-shot learning requires a classifier to recognize samples from classes that were not observed during training. Such characteristic makes it especially suitable for processing and analyzing social media data, as social media data is mostly unlabeled and it is difficult to label a good amount of data representative of various classes.

With the unprecedented volume of data generated from CPSS, the use of advanced graph-based methods, e.g. graph embedding and graph neural network, to model the relationship between data items has become a promising research direction. Although many studies have been conducted, research on how to efficiently and effectively exploit

the power of graph-based methods, given the overwhelming amount of CPSS data, is still at an early stage. Recently, research on how to effectively utilize existing, quality knowledge bases within DNNs has attracted significant attention [206]. The knowledge stored in many existing Knowledge Base (KB) and Knowledge Graph (KG) represents facts and human wisdom accumulated over centuries. Including such knowledge in learning systems has great potential. On one hand, systems do not need to learn existing knowledge from scratch; on the other, mistakes made in classification previously can be avoided to a great extent. Embedding has emerged as an important approach to prediction, inference, data mining, and information retrieval. Graph embedding algorithms that represent the hierarchical structure of a knowledge base in the format of vectors have been increasingly investigated. By transferring the rich structural information from knowledge bases to learning systems, better prediction, classification and recommendation performance could be anticipated. However, the convergence of deep learning and knowledge graph embedding is a challenging research topic that has not been extensively studied.

For the issues discussed above, we propose a new zero-shot learning method which exploits the use of existing knowledge graphs for the classification of large amounts of social text data (i.e. Twitter messages related to COVID-19) without training data. Following the key ideas in zero-short learning, the proposed method does not explicitly define the class labels. The pre-trained sentence based BERT model Sentence-BERT (S-BERT) [139] is first used to represent tweet messages in the embedding space to be further matched with classes. As the purpose of S-BERT is to learn a sentence-level representation, while most class labels contain only one or few words, S-BERT embedding may not be as semantically consistent as word-level embedding methods. To address this problem, we construct a knowledge graph embedding model for label representation with a comprehensive knowledge graph named ConceptNet [159]. The sentence embedding is then projected to the knowledge graph through the least-squares linear projection. The proposed model is referred to as the S-BERT-KG model. We apply the model to COVID-19 related tweet classification without any labeled data for training. To our best knowledge, this is the first work that takes the zero-shot learning architecture for tweet classification. Experimental results demonstrated that the proposed S-BERT-KG model is able to gain significant improvement over other baseline models and produce reasonable prediction accuracy without any labeled data.

We review some representative work on social media data classification, zero-shot text classification, graph neural networks, and graph embedding in Section 6.2.1. In Section

6.2.2, we describe the SBERT-KG architecture, knowledge graph embedding, and the zero-shot text classification procedure in detail. Section 6.2.3 describes the experiments conducted on a large Twitter dataset related to COVID-19 and presents the evaluation results compared with baseline models.

6.2.1 Related Work

Social media data classification: As of May 2020, there are around 500 million tweets posted on Twitter each day. Social media "senses" nearly everything happening around the world, and the produced data has become a valuable source for research in different disciplines. In contrast to data collected from the physical world, social media data has some attractive features. For example, it covers far more areas and topics, can be collected at low cost, and enjoys high-level semantics understandable to human users. With the ideas of 'citizen sensing' and 'human as sensors', a number of real-world applications have been developed, e.g. traffic event detection [37], spammer detection [62], and natural disaster assessment [115, 76]. Various NLP techniques have been applied to extract useful information from short tweet messages. For instance, the work in [76] proposed a large word2vec embedding that was trained on 52 million crisis-related tweets and used to classifies crisis tweets using Support Vector Machines, Naive Bayes and Random Forest methods. The study in [115] designed a framework based on the Convolutional Neural Network (CNN), which used convolution and pooling operations to capture important information to identify useful or crisis-related tweets. The work in [62] developed a collaborative neural network spammer detection mechanism, which fused multi-source information by collaboratively encoding long-term behavioral and semantic patterns. The work in [37] implemented CNN and Recurrent Neural Network (RNN) deployed on the top of word-embedding models for detecting traffic events. Recently, Twitter has seen a massive surge in the daily traffic related to the COVID-19 pandemic due to the COVID-19 outbreak. A number of studies have collected a large amount COVID-19 related social media data [135] and applied topic modeling techniques [121] to analyze topic trends. In short of labeled data, studies exploiting supervised or semi-supervised NLP models to automatically classify tweets are still very limited. In this paper, we apply state-of-the-art NLP techniques to extract knowledge from COVID-19 related messages on social media in an unsupervised manner.

Zero-shot text classification: To extract knowledge from Cyber-Physical-Social (CPS) data, traditional supervised learning paradigm needs sufficient labelled data for

model training, while most CPS data is not labeled. Recently, the research focus has shifted towards learning from a large amount of unlabeled data, and has achieved remarkable progress in domain adaptation [49], few-shot learning [155], zero-shot learning [78], or more generally, transfer learning [143]. Bidirectional Encoder Representations from Transformers (BERT) [42] and its variants (e.g, S-BERT [139], RoBERTa [98], BART [90]) are pre-trained language models that achieved state-of-the-art results in various text classification tasks, e.g., machine translation, question answering and language inference. While in the zero-shot classification scenario, a classifier is required to work on labels that it is not explicitly trained with. In the computer vision domain, one common approach for zero-shot learning is to use existing feature extractors to represent images and any possible label names in their corresponding embedding space [210]. Some annotated data might be used to align the image and label embedding. The framework allows any label (seen or unseen during training) and any image to be embedded in the same latent space to measure their similarity. In the text domain, one single NLP model can be utilized to embed both data and any class names into the same embedding space without an alignment step. In particular, [134] concatenated both the sentence data and class names as model input, and treated zero-shot learning as a binary classification task; [198] took both sentence and label names into the BERT model and considered zero-shot text classification as a Natural Language Inference (NLI) task. Inspired by the work in [139] and [198], we propose a zero-shot text classification architecture for social media data analysis. The proposed method directly makes use of models pre-trained with NLI tasks, thus does not need any labeled data for model training.

Graph neural network and graph embedding: The growing number of connected IoT equipments can be described with graphs, e.g. vertices denote sensors while edges denote the connection between them. Understanding the structure of such complex and ubiquitous IoT networks remains a challenging task. Graph Neural Network (GNN) [82], a class of deep learning method designed to perform inference on data described by graphs, motivated researchers to model IoT data based on the internal relationships between different sensors. One typical application of GNNs in IoT is traffic prediction, where the traffic network can be modeled as a spatial-temporal graph; the nodes are loop sensors installed on roads, and the edges represent the intersections or road segments connecting these sensors. The work in [213] and [36] utilized Graph Convolutional Networks (GCNs) to capture spatial dependency, leveraged recurrent neural networks (RNNs) for modeling temporal dynamics, and attained state-of-the-art performance in the traffic prediction

task. With the enormous amount of cyber data available on the Internet, significant efforts from industry and academia have been made into constructing knowledge bases [206], e.g. DBpedia [89] and ConceptNet [159]. To represent the hierarchical structure of a knowledge base in the format of vectors, graph embedding methods have been increasingly investigated, e.g. DeepWalk [130], Node2vec [58], SDNE [181], etc. Meanwhile, recent studies [209, 96] leveraged these external knowledge graphs to further enhance language representation by integrating the rich structured knowledge facts into NLP model input. In this paper, we further explore the knowledge graph embedding technique in the zero-shot text classification scenario for better natural language understanding.

6.2.2 Zero-shot Learning with Knowledge Graph

In this section we describe the S-BERT-KG model in terms of a tweet classification task for extracting informative tweets. Let X be the set of tweets to be categorized and S be the set of possible class names (or labels) for X. The goal is to represent both tweets X and label names S in the same embedding space, so as to classify any tweets by measuring the similarity between tweet embeddings (usually in a form of a sentence) and label embeddings, without using any labeled data. It should be noted that the set of possible label names S are not explicitly defined in the zero-shot scenario, which can be any object in a given knowledge graph or words in a given vocabulary. As it is impractical to evaluate the model on all possible labels in a vocabulary, we need to specify a possible label set, i.e., seven representative classes of COVID-19 tweets in this study.

The overall architecture of S-BERT-KG is shown in Figure 6.2. A pre-trained Sentence BERT (S-BERT) [139] is firstly used to embed sentence u and label v where $x \in X$ and $y \in S$. An external knowledge graph (i.e. ConceptNet) is used to construct a knowledge graph embedding space with the retrofitting [47] method. Sentence embedding and label embedding are then projected to the knowledge graph embedding space via a learned projection matrix P to determine if a tweet belongs to a specific class(es) based on their cosine similarity. In the following, we firstly introduce the relevant BERT models, then present the method used for knowledge graph embedding, and finally show the zero-shot text classification process.

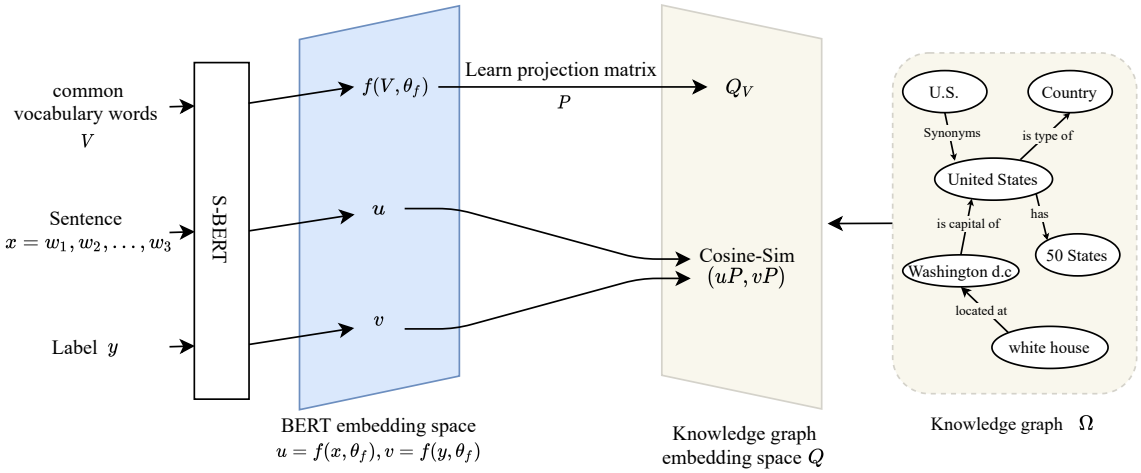


Figure 6.2: S-BERT with Knowledge Graph embedding (S-BERT-KG) architecture

Sentence BERT (S-BERT)

BERT has achieved state-of-the-art performance in a number of NLP tasks, e.g. text classification, question answering, and language inference. However, for sentence-pair regression tasks such as language inference, BERT requires that both sentences are fed into the network simultaneously, which demands significant computation at resources especially when the number of sentences is large. The recent Sentence-BERT architecture [139], that uses the Siamese structure to derive semantically meaningful sentence embeddings, reduces the effort for finding the most similar pair while maintaining accuracy. For zero-shot classification purposes, instead of taking sentence pairs, the S-BERT model takes a Sentence x and a Label y into the model to measure their similarity. The architecture of the S-BERT model is illustrated in Figure 6.3.

As shown in Figure 6.3, a tokenizer is used to split the tweet input into tokens $\{w_1, w_2, ..., w_n\}$, add special tokens (e.g., [CLS] and [SEP]) and convert these tokens into indices of the tokenizer vocabulary. The output hidden state C is used as the aggregated sequence representation for the classification tasks and $\{t_1, t_2, ..., t_n\}$ represents the corresponding word embedding vectors for tokens $\{w_1, w_2, ..., w_n\}$. S-BERT contains an additional pooling layer to evaluate models under different pooling strategies by using: 1) the output of CLS-token (C); 2) the mean of all hidden states, or 3) the maximum of all output vectors. It is trained with SNLI [22], Multi-NLI [190], and STS [25] datasets so as to provide similar

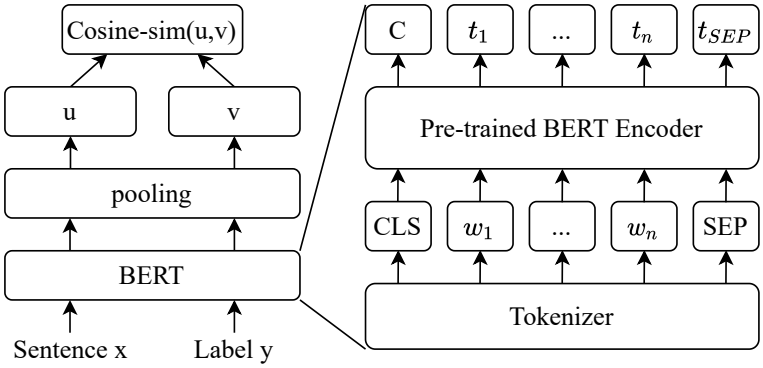


Figure 6.3: Sentence-BERT (S-BERT) architecture

sentence embeddings for semantically similar sentences [139]. On the basis of this idea, we could directly take the pre-trained model and classify a sentence x to a label $\hat{y}$ with Eq. 6.6, given all possible label names S.

$$\begin{aligned} \hat{y} &= \arg\max_{y \in S} \cos(u, v) \\ u &= f(x, \theta_f), x \in X \\ v &= f(y, \theta_f), y \in S \end{aligned} \tag{6.6}$$

where *cos* is the cosine similarity, f is the function represented by S-BERT, θ_f denotes the pre-trained parameters of f, X and S represent the set of tweets and possible class labels respectively.

Knowledge Graph Embedding

One issue using S-BERT for zero-shot text classification is that S-BERT is trained to learn effective sentence-level representations, but may not generate semantically consistent single word label representations as other word embedding methods do (e.g. word2vec and GloVe). Also, all pre-trained language models (e.g. BERT, word2vec, and GloVe) lack common-sense or domain-specific knowledge, which usually results in unsatisfactory performance for short message representation. To address these two issues, we construct a knowledge graph embedding model based on ConceptNet [159] for zero-shot tweet classification.

ConceptNet [159] is a knowledge graph that connects words and phrases of natural language with labelled edges. Its knowledge is collected from multiple sources that include expert generated resources and crowd-sourcing. It aims to represent the common sense knowledge involved in understanding language and to improve natural language applications by enabling applications to better understand the meaning behind the words. ConceptNet represents relations and words (e.g. shown in Fig. 6.2) as triples, e.g. (Washington d.c., capitalOf, United States), (United States, has, 50 States) and (United States, typeOf, Country).

By adding the rich graph structure information from ConceptNet into common NLP techniques, particularly, word embedding methods (i.e. word2vec and GloVe), we could construct a semantic space that is potentially more effective than distributional semantics in terms of better prediction, classification and recommendation performance. Retrofitting [47] is a process that adjusts an existing matrix of word embeddings q with a knowledge graph Ω. It calculates new word vectors q_i with the objective of staying close to both their original values in other word embedding vectors (i.e. word2vec and GloVe) $\hat{q}_i$, and their neighbors q_j in the knowledge graph with edges $(i, j) \in E$. The knowledge graph embedding Q can be computed by minimizing the following objective function:

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \left\| q_i - \hat{q}_i \right\|^2 + \sum_{(i,j) \in E} \beta_{ij} \left\| q_i - q_j \right\|^2 \right] \tag{6.7}$$

where $Q = (q_1, q_2, ..., q_n)$ is the learned knowledge graph embedding matrix, α and β values control the weights of word embedding and knowledge graph. The word vectors in Q are firstly initialized to be equal to the vectors in $\hat{Q}$ and then retrofitted. An iterative updating method [21] is used to calculate Q that converges in just a few iterations.

Embedding Alignment and Classification Process

We have embedded tweet and label representations into the S-BERT embedding space, and constructed a knowledge graph embedding for all possible labels. Next, we need to learn a linear projection function that can align the tweet and label embeddings. This allows one to embed any tweets and labels into the same knowledge embedding space to measure their similarity. For text data, this process can be done by aligning the representations of the same vocabulary words in different embedding space. We take the top K frequently used English words from the vocabulary of the S-BERT model, and learn a projection

matrix P with least-squares linear projection. The process maps the word embeddings from the S-BERT embedding space to the knowledge graph embedding space. For label representations, whether using the representation after projection $f(y, \theta_f)P$, or directly deploying the embedding from knowledge graph embedding space Q_y, leads to similar prediction performance. After this embedding alignment process, the equation to classify a given tweet becomes:

$$\hat{y} = \arg\max_{y \in L} \cos(f(x, \theta_f)P, f(y, \theta_f)P) \tag{6.8}$$

The embedding alignment and the zero-shot text classification of using the S-BERT-KG method are then detailed in Algorithms 4 and 5 respectively. The purpose of the embedding alignment process is to learn a projection matrix P that maps the S-BERT representation to the knowledge graph embedding space. In Algorithm 4, we first learn a knowledge graph embedding using ConceptNet (Line 1), then obtain the representation of the most common vocabulary words from S-BERT and the knowledge graph embedding (Lines 2-5), and finally learn a projection matrix (Line 6). For zero-shot text classification (as shown in Algorithm 5), we use the tweet representations (Lines 6-7) and the label representations (Lines 2-3) to generate label prediction (Line 8) without any training process.

Algorithm 4 S-BERT-KG embedding alignment

Input: pre-trained S-BERT model parameters θ_f, GloVe word embedding $\hat{Q}$, knowledge graph Ω

Output: Projection Matrix P

1: Calculate knowledge graph embedding Q with GloVe word embedding $\hat{Q}$ and knowledge graph Ω using Eq. 6.7;
2: Initialize S-BERT model with pre-trained parameters θ_f;
3: Take K most common vocabulary words V from the S-BERT model;
4: Obtain knowledge graph embeddings for words V: Q_V;
5: Obtain their S-BERT embeddings for words V: $f(V, \theta_f)$;
6: Learn a least-squares linear projection matrix P from $f(V, \theta_f)$ to Q_V.

6.2.3 Experiments and Evaluation

In this section, we first describe the large unlabeled COVID-19 Twitter dataset we collected from Twitter and the two small labeled datasets we manually created for the evaluation

Algorithm 5 S-BERT-KG zero-shot text classification

Input: pre-trained S-BERT model parameters θ_f, unlabeled tweet dataset X, possible label names L, projection matrix P

Output: label predictions $\hat{Y}$.

1: **for** each label y in L **do**
2: Represent each label y with S-BERT embedding: $f(y, \theta_f)$;
3: Project $f(y, \theta_f)$ into the knowledge graph embedding space with projection matrix P, which gives $f(y, \theta_f)P$;
4: **end for**
5: **for** each tweet x in X **do**
6: Represent each tweet x from X with S-BERT embedding: $f(x, \theta_f)$;
7: Project $f(x, \theta_f)$ into the knowledge graph embedding space with projection matrix P, which gives $f(x, \theta_f)P$;
8: Generate label prediction $\hat{y}$ for tweet x with Eq. 6.8;
9: **end for**

purpose. We then report the evaluation results and compare the performance of S-BERT-KG with several state-of-the-art deep learning methods.

Dataset

Table 6.3: Distributions of the COVID twitter datasets

Classes	Multi-class Dataset (D1)	Multi-label Dataset (D2)
Advice	137	155
China	449	554
Mask	225	272
News	309	408
Transportation	46	57
USA	476	596
Vaccine	96	115
total	1,738	1,941

A number of recent studies have been conducted to process COVID-19 related tweets, e.g. sentiment analysis and topic modeling. However, research that focuses on extracting informative tweets and classifying them to meaningful classes has rarely been found due to the lack of a labeled dataset. Moreover, we have not spotted any public COVID-19 related

twitter datasets that are labeled with meaningful categories. We collected all COVID-19 related tweets with tweet IDs provided by GeoCoV19 [135], which contain more than 524 million multilingual tweets from Feb 1st to May 1st 2020. We took one week to collect all the tweets with IDs in GeoCoV19 using the Twitter API, and formed a 12.8GB dataset. So far, we have not seen any notable research on zero-shot learning with unlabeled Twitter datasets. In this study, we only used the tweets written in English, and preprocessed the tweet text to lower case and removed punctuation, question marks and/or URLs.

To evaluate the performance of the proposed model using standard evaluation metrics for supervised learning, we manually labeled some tweets and constructed two small datasets. Hashtags represent keywords or topics in a tweet message. To define meaningful categories for the datasets, we collected frequently used hashtags from the COVID-19 twitter dataset (e.g., #COVID19, #socialdistancing, #wuhan, #covid19_US, #vaccine, #n95, etc.). As it was impractical to define very detailed categories (e.g., COVID19 in US, COVID19 in UK, COVID19 in China, COVID19 in Italy, COVID19 in Spain, etc.) to label all these tweets, we manually examined the hashtags and selected seven trending topics related to the COVID-19 pandemic. It should be noted that the selected seven labels are exploited only for the evaluation purposes. We could select any word from a knowledge graph vocabulary as a label for zero-shot text classification. For each day between Feb 1st and May 1st, we sampled a number of tweets and categorized them into the following seven classes.

1. **Advice**: Stay at home, wash hands, wear mask or social distancing.
2. **China**: Wuhan, China Coronavirus Updates, China news, or other tweets related to China.
3. **Mask**: Mask shortage, wear mask, mask types, etc.
4. **News**: Coronavirus updates, news, rules, etc.
5. **Transportation**: Flights, traffic, traveling, etc.
6. **USA**: U.S. Coronavirus Updates, U.S. news, or other tweets related to the United States.
7. **Vaccine**: Vaccine news, vaccine progress, vaccine injection, etc.

One labeled dataset is for evaluation of multi-class classification, and the other for multi-label classification as one tweet may be related to multiple topics. After removing non-informative and duplicate tweets, the two datasets (D1 and D2) contained 1,738 and 1,941 labeled tweets respectively. D2 contains all the tweets in D1 and an additional 203 tweets with multiple labels. The detailed distribution of the datasets is shown in Table 6.3.

For the experiments we used the ConceptNet 5.7, which contains over 21 million edges and over 8 million nodes (around 1.5 million English nodes). We constructed a sub knowledge graph from ConceptNet using its API with all the vocabulary words that appearing in the COVID-19 Twitter dataset. However, we found that the constructed sub-graph embedding did not provide any improvement over the original version of ConceptNet Numberbatch[1]. Therefore, we utilized the original version in our proposed architecture for better reproducibility.

Baseline Models

We re-implemented six baseline models for zero-shot multi-class and multi-label classification, and compared their performance with the proposed S-BERT-KG model.

- **GloVe-AVG**: We used GloVe [127] word embedding to represent each word in a sentence and all the possible label names. The averaged embedding vector was used to represent the sentence and further measure distance with each label.
- **BERT-CLS and BERT-AVG**: We used the last hidden state output of the standard BERT model (described in Section 2.3.4) to represent sentence embedding and label embedding. The output of the CLS token was used for the BERT-CLS version; and the averaged last hidden state output was used for the BERT-AVG version.
- **S-BERT**: We used Sentence-BERT [139] pre-trained with SNLI, MultiSNL and STS datasets to represent sentence and label embeddings for zero-shot classification.
- **S-BERT-GloVe**: Besides using the S-BERT model, We also learned a function to project S-BERT embedding into the GloVE embedding space.
- **BART-NLI**: The study presented in [198] showed that zero-shot text classification can be modeled in a natural language inference architecture, where the hypothesis is

[1]https://github.com/commonsense/conceptnet-numberbatch

constructed by associating a label, e.g., "news", with the pre-defined problem "The text is about ?". Given a sentence, the model is trained to determine whether the hypothesis is true. A recent BART model [90], pre-trained with the SNLI dataset, was used in our zero-shot tweet classification task for comparison.

In this study, we chose a traditional word embedding model (GloVe-AVG) and five deep learning based models for performance evaluation. We show that the zero-shot text classification task can be modeled as a natural language inference problem by comparing BERT-CLS and BERT-AVG with S-BERT. We compare S-BERT and S-BERT-GloVe with the proposed S-BERT-KG to demonstrate the effectiveness of incorporating an external knowledge graph for better language understanding. We also show the performance of the proposed model against the state-of-the-art BART-NLI model.

Setup

The GloVe [127] word embedding vectors used in the experiments contain 400K uncased words in the vocabulary and were pre-trained with 6 billion tokens[2]. We use the GloVe word vector with 300 dimensions for all experiments.

For the BERT-CLS and BERT-AVG models, we used the pre-trained BERT-uncased-base model, which contained 12 transformer blocks and 768 dimensions in the hidden units. For the S-BERT, S-BERT-GloVe and the proposed S-BERT-KG models, we took the large S-BERT model pre-trained with the SNLI and STS datasets as described in [139]. The "mean" strategy for the pooling layer was applied.

For the S-BERT-GloVe and S-BERT-KG, we selected 20,000 most frequently used words from the BERT vocabulary to learn the projection matrix P. For BART-NLI, we engaged the large BART model pre-trained with the SNLI dataset. We applied the transformers[3] and sentence-transformers[4] libraries to implement all the models; all pre-trained models used in our study can be downloaded via these two libraries.

The experiments were run using PyTorch 1.7.1, Tensorflow 2.4, Python 3.6, and Windows 10 running on a desktop computer with an i7-9700F CPU, 32GB RAM and RTX-2070S GPU.

[2]https://nlp.stanford.edu/projects/glove/
[3]https://huggingface.co/transformers/
[4]https://www.sbert.net/

Evaluation

We exploited the standard evaluation metrics for assessing classification performance, i.e. accuracy, weighted average precision, recall and F1, with the two labeled datasets. For the multi-label classification task, we calculated the exact match for accuracy and also reported Hamming loss, which is the fraction of labels that are incorrectly predicted. The evaluation results of the proposed S-BERT-KG and other models are shown in Table 6.4 and 6.5 and 6.6.

Table 6.4: Performance Comparison of Different Models for All Seven Labels in Multi-class Classification in Terms of Precision (P), Recall (R), and F1 Score (F1)

Labels	BERT-AVG			S-BERT			S-BERT-GloVe			S-BERT-KG		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Advice	45.53	40.88	43.08	41.67	47.45	**44.37**	36.21	15.33	21.54	44.05	27.01	33.48
China	63.84	25.17	36.10	92.59	55.68	69.54	91.88	55.46	69.17	91.26	90.65	**90.95**
Mask	100.0	04.89	09.32	73.64	78.22	**75.86**	79.52	58.67	67.52	74.75	67.11	70.73
News	21.57	97.73	35.34	56.82	08.09	14.16	67.14	15.21	24.80	68.93	45.95	**55.15**
Transportation	00.00	00.00	00.00	68.42	28.26	40.00	93.75	32.61	**48.39**	80.00	26.09	39.34
USA	100.0	05.67	10.74	82.65	17.02	28.22	68.63	29.41	41.18	68.65	53.36	**60.05**
Vaccine	00.00	00.00	00.00	09.87	93.75	17.86	09.76	96.88	17.73	20.72	89.58	**33.66**
macro avg	47.28	24.91	19.23	60.81	46.92	41.43	63.84	43.37	41.47	64.05	57.11	**54.77**
weighted avg	64.25	29.29	23.15	71.83	40.28	43.58	70.64	40.10	46.25	71.04	62.66	**64.44**

Table 6.5: Performance Comparison of Different Models for All Seven Labels in Multi-label Classification in Terms of Precision (P), Recall (R), and F1 Score (F1)

Labels	BERT-AVG			S-BERT			S-BERT-GloVe			S-BERT-KG		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Advice	23.38	41.94	30.02	34.23	32.90	**33.55**	23.33	09.03	13.02	25.00	05.81	09.42
China	50.38	24.19	32.68	95.54	61.91	75.14	90.62	62.82	74.20	95.59	90.07	**92.75**
Mask	82.86	10.66	18.89	83.63	69.49	**75.90**	89.51	47.06	61.69	96.46	40.07	56.62
News	25.84	58.58	35.86	49.28	08.33	14.26	42.72	10.78	17.22	60.00	31.62	**41.41**
Transportation	12.50	01.75	03.08	34.78	14.04	**20.00**	66.67	07.02	12.70	71.43	08.77	15.62
USA	62.79	13.59	22.34	75.51	18.62	29.88	68.35	25.00	36.61	64.40	60.40	**62.34**
Vaccine	14.29	01.74	03.10	10.86	91.30	19.41	11.65	96.52	20.79	18.84	87.83	**31.03**
macro avg	38.86	21.78	20.85	54.83	42.37	38.31	56.12	36.89	33.75	61.68	46.37	**44.17**
weighted avg	48.40	25.54	26.14	69.23	38.99	43.80	65.59	37.00	42.59	70.55	56.19	**58.76**

We report the precision, recall and F1 score of using the 4 BERT-based models for all seven different labels selected in our studies in Table 6.4 (for multi-class classification) and 6.5 (for multi-label classification). From both tables, we observe the proposed S-BERT-KG model significantly outperformed all other BERT-based models for labels: China, News,

Table 6.6: Performance Comparison of Different Models in Terms of Accuracy (A), Precision (P), Recall (R), F1 Score (F1), Hamming Loss (H), and Running Time (t) in Seconds

Methods	Multi-class Classification					Multi-label Classification					
	A	P	Rl	F1	t	A	P	R	F1	H	t
GloVe-AVG	52.76	65.15	52.76	46.93	**2.76**	32.05	75.96	40.10	42.73	12.93	**2.79**
BERT-CLS	30.96	50.26	30.96	23.83	10.84	3.76	24.46	37.92	28.76	32.73	11.42
BERT-AVG	29.29	64.25	29.29	23.15	10.85	10.20	48.40	25.54	26.14	19.95	11.24
S-BERT	40.27	71.83	40.28	43.58	14.59	17.88	69.23	38.99	43.80	17.77	15.64
S-BERT-GloVe	40.10	70.64	40.10	46.25	30.67	18.70	65.59	37.00	42.59	17.87	31.92
BART-NLI	51.21	**80.53**	51.21	50.41	107.19	**37.10**	**84.78**	39.31	41.94	14.17	116.75
S-BERT-KG	**62.66**	71.04	**62.66**	**64.44**	30.25	34.00	70.55	**56.19**	**58.76**	**12.67**	31.19

USA and Vaccine. For example, in the multi-class classification scenario (Table 6.4), the S-BERT-KG model improved by around 21% F1 for tweets related to China, 20% F1 for tweets related to news, 19% F1 for tweets related to USA, 16% F1 for tweets related to vaccine, 13.30% in macro averaged F1 and 18.19% in weighted average F1. This confirmed that projecting S-BERT embedding to the knowledge graph embedding space could, in general, offer better performance for zero-shot classification.

One highlight from Table 6.6 is that the BERT-CLS and the BERT-AVG models had the lowest performance in both classification scenarios. It shows that directly using the BERT model was not suitable for sentence classification without any training data. While applying the S-BERT model pre-trained with SNLI, MultiNLI and STS datasets, we observed significant improvement in terms of all evaluation metrics, e.g. around 10% in accuracy and 20% in F1 score in the multi-class scenario compared with BERT-CLS. This observation demonstrated the effectiveness of sentence representation with large S-BERT models pre-trained with Natural Language Inference (NLI) datasets. Table 6.6 shows that the GloVe-AVG model outperformed the S-BERT with 12.49% in accuracy for multi-class classification and 14.17% in accuracy for multi-label classification, indicating that although S-BERT could learn good sentence-level embeddings, it might not generate semantically consistent word-level embeddings for labels.

Another notable observation in Table 6.6 is that the proposed S-BERT-KG model significantly outperformed S-BERT-GloVe with respect to all the metrics considered. This indicated that the knowledge graph embedding could integrate common sense knowledge from the external knowledge base and improve model performance. When comparing S-BERT-KG with the recent BART-NLI model, we also observed significant improvement with 10.76% in accuracy, 10.76% in recall, 13.45% in F1 for multi-class classification, and

16.88% in recall, 16.82% in F1, 2.50% in hamming loss for multi-label classification. It should be noted that the BART-NLI model is based on a recent seq2seq architecture with a bidirectional encoder (e.g. BERT) and a left-to-right decoder (e.g. GPT), which outperformed BERT in NLI tasks.

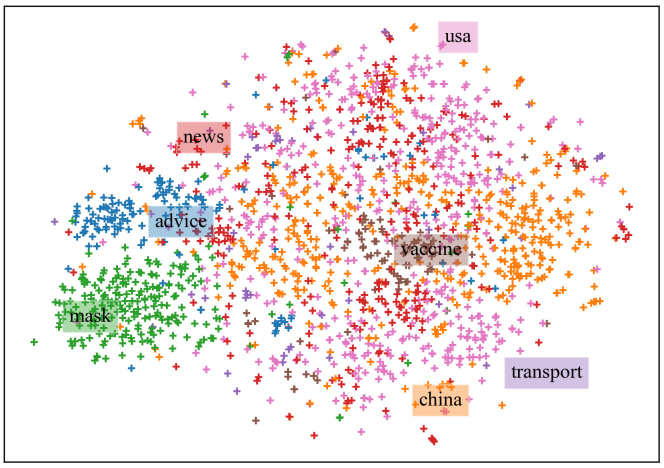
In Table 6.6, we also reported the running time (t) of different models for multi-class classification and multi-label classification. The GloVe-AVG model that directly calculates the average value of word embedding vectors took the least time for prediction. If we compare BERT-CLS, S-BERT and BART-NLI, the increased time was related to the size of the models. For multi-class classification, the S-BERT model took 14.59 seconds while the S-BERT-KG model took 30.25 seconds. Thus, knowledge graph embedding alignment process consumed around 15 seconds to run. It is also noted that our proposed model shows a clear superiority to the BART NLI model in both efficiency and effectiveness. Since the S-BERT-KG model does not require a training process, which may take hours or days, it has the potential to be applied to real-time or near real-time streaming tweet classification.

For further comparison, we generated the t-SNE visualization[5] of the sentence and label embeddings (as shown in Figure 6.4) of the GloVe-AVG, S-BERT, and S-BERT-KG models. In the sub-figures, different colors represent different labels. By comparing 6.4a with 6.4b we can observe that the S-BERT model generates better sentence embeddings, i.e. the tweets related to China are better clustered. However, as the labels are poorly aligned in S-BERT, the overall performance was worse than GloVe-AVG as shown in Table 6.6. While using the S-BERT-KG, labels appeared much closer to their corresponding data clusters compared to the S-BERT model, and the sentences were also well clustered. With the visualization, we could further confirm that combing the sentence embedding model with the knowledge graph embedding is an effective method for leveraging unlabeled data for zero-shot tweet classification.

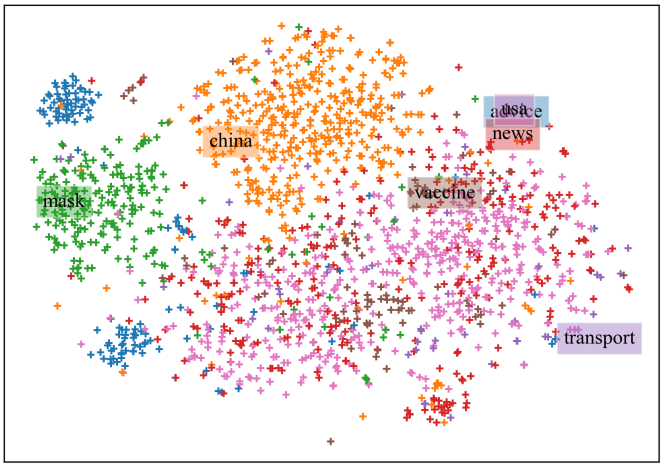
6.3 Summary and Discussion

In this chapter, we focused on fine-grained text classification tasks with limited amount of labelled data from social media. We present two smart city applications: Domain Adaptation for Crisis-related Data and Zero-shot Text classification for COVID-19 Tweet

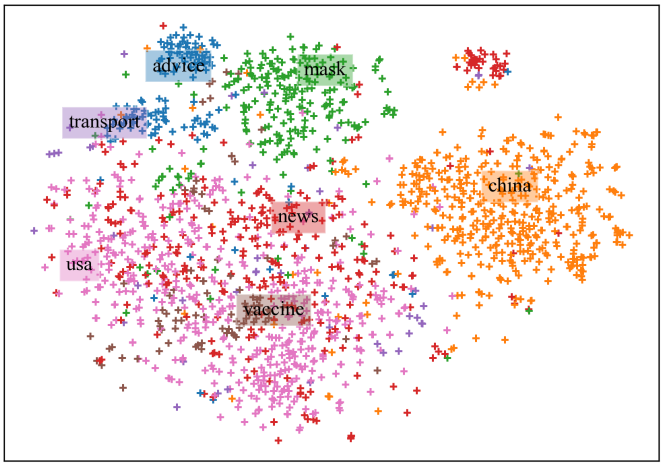
[5]https://lvdmaaten.github.io/tsne



(a) t-SNE visualization of averaged GloVe embeddings



(b) t-SNE visualization of Sentence-BERT embeddings



(c) t-SNE visualization of embeddings with Sentence-BERT to Knowledge Graph Projection

Figure 6.4: t-SNE visualization of sentence and label embeddings of different methods

in Section 6.1 and Section 6.2 respectively.

Domain Adaptation for Crisis-related Data: During crisis events, users may post informative tweets about affected individuals, utility damage or cautions on social media platforms. If such tweets are efficiently and effectively processed and analysed, city organizations may gain a better situational awareness of the affected citizens and provide better response actions. Advances in deep neural networks have significantly improved the performance in many social media analysing tasks, e.g., sentiment analysis, fake news detection, crisis data classification, etc. However, deep learning models require a large amount of labeled data for model training, which is impractical to collect, especially at the early stage of a crisis event. To address this limitation, we proposed a BERT-based Adversarial Domain Adaptation model (BERT-ADA) for crisis tweet classification. A pre-trained BERT model was utilized to extract valuable knowledge or features from tweets. Since the amount of labeled data is usually limited at the early stage of natural disasters, we leverage the adversarial domain adaptation technique to classify data in an unsupervised manner. The evaluation results in Section 6.1.3 clearly showed the advantages of the proposed BERT-ADA over other baseline models.

Zero-shot Text classification for COVID-19 Tweet: COVID-19 pandemic has become a hot topic on Twitter. In this chapter, we explore the automatic classification of COVID-19 related tweets on social media without any labelled data for training using the zero-shot learning technique. Deep learning models typically require a large amount of labeled data for model training, while most CPSS data is not labeled, making it impractical to build effective learning models using traditional approaches. In addition, the current state-of-the-art, pre-trained NLP models do not make use of existing knowledge graphs, thus often leading to unsatisfactory performance in real-world applications. Our study managed to address these two issues and develops the S-BERT-KG model following the zero-shot learning paradigm for classification of COVID-19 related tweets. Performance of the S-BERT-KG model has been both impressive and promising, as evidenced by the evaluation results on both multi-class and multi-label classification tasks in Section 6.2.3. For future work, we plan to refine the proposed model in a number of directions. As we did not find more recent models pre-trained in the Sentence BERT architecture, we engaged the S-BERT model described in [139] for all the experiments and evaluation. It is expected that the S-BERT-KG model could be further improved with more recent models, e.g. roBERTa [98] and BART [90]. We plan to investigate the self-training method to further exploit the knowledge from the large amount of unlabelled data, and exploit the

few-shot learning technique when only a limited amount of labeled data is available. With the proposed zero-shot text classification architecture, we would like automatically create more labeled data for more comprehensive evaluation. Currently, all the labels used in this study are individual words. However, it may destroy its original semantics by representing key phrases with individual words using word embedding methods. We will further explore the performance of knowledge graphs in addressing this issue, and further apply the power of knowledge graphs, graph embeddings, and graph neural networks to other social IoT applications.

Chapter 7

Conclusion and Future Work

Yesterday is history, tomorrow is a mystery, today is God's gift, that's why we call it the present.

-Joan Rivers

This chapter concludes the thesis with a review of the work presented in previous chapters, and summarises the research contributions. Furthermore, it discusses and suggests directions for future work.

7.1 Conclusions

The thesis provided a comprehensive review for deep learning and smart city data processing, demonstrate applications in intelligence transpiration and social media analysis domain, and proposed solutions for issues related to real-world smart city applications which have not been sufficiently addressed in the literature.

A general introduction for deep learning is presented in Chapter 2. This chapter started off by introducing three basic types of machine learning , which are supervised learning, unsupervised learning and semi-supervised learning. Then, training and testing processes, which includes objective function, regularisation and evaluation metrics, were introduced. Next, We focused on the four most widely used deep neural network models: Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Generative Adversarial Network (GAN), and Bidirectional Encoder Representations from Transformers (BERT).

We showed how these models could be used in smart city applications and briefly discussed how to select models for different smart city applications. Finally, some challenges and future directions of deep learning from smart city data were discussed.

A overview about smart city data processing in presented in Chapter 3. This chapter started off by introducing the background of smart city and the framework of smart city data processing. Then, different city data representations (time series, image and video, and text) are listed. Next, We focus on the four representative application domains for which deep learning methods have been applied and demonstrated notable performance improvement, i.e., Transportation, Healthcare, Environment and Public safety. Finally, some future directions of smart city applications are discussed.

In Chapter 4, we showed extracting knowledge from sensory data with a traffic flow prediction task. Traffic flow prediction is an important while complex problem in transportation modeling and management. Many uncertain, non-linear and stochastic factors could have large influence on the prediction performance. Existing studies still have some issues unaddressed, e.g., the models only predict the traffic flow at next time step while travelers may need a sequence of predictions to make better, long-term decisions; temporal factors are (e.g., day of the week, national holiday) usually not well considered during prediction. To address these limitations, We proposed an attention-based recurrent neural network architecture for multi-step traffic flow prediction. Experimental results demonstrate that the proposed method has superior performance compared to the existing models. We also show how the method can be used to develop traffic anomaly detection systems.

In Chapter 5, we presented multi-modal deep learning models for a traffic event detection task and a crisis-related data classification task. For traffic event detection, we used semi-supervised deep learning with data of different modalities, e.g., physical sensor observations and social media data. Unlike most existing studies focusing on data of single modality, the proposed method makes use of data of multiple modalities that appear to complement and reinforce each other. Meanwhile, as the amount of labelled data in big data applications is usually extremely limited, we extend the multi-modal Generative Adversarial Network model to a semi-supervised architecture to characterise traffic events. We evaluated the model with a large, real-world dataset consisting of traffic sensor observations and social media data collected from the San Francisco Bay Area over a period of four months, and demonstrate the advantages of the proposed model in extracting and classifying traffic events. For crisis-related data classification, we proposed an adversarial training method to process multi-modal data as well as improve the robustness of deep

learning models. The evaluation results clearly demonstrate the advantages of the proposed model in improving the robustness of tweet classification.

In Chapter 6, we presented fine-grained text classification for social media data in crisis-related data classification and COVID-19 tweet classification tasks. Deep learning models typically require a large amount of labeled data for model training, while most CPSS data is not labeled, making it impractical to build effective learning models using traditional approaches. To address this limitation, we proposed a BERT-based Adversarial Domain Adaptation model (BERT-ADA) for crisis tweet classification. Our experiments with three real-world crisis datasets demonstrate the advantages of the proposed model over several baselines. In addition, the current state-of-the-art, pre-trained Natural Language Processing (NLP) models do not make use of existing knowledge graphs, thus often leading to unsatisfactory performance in real-world applications. To address the issues, we propose a new zero-shot learning method which makes effective use of existing knowledge graphs for the classification of very large amounts of social text data. Experiments were performed on a large, real-world tweet dataset related to COVID-19, the evaluation results show that the proposed method significantly outperforms six baseline models implemented with state-of-the-art deep learning models for NLP.

7.2 Research Findings

After the brief summary above, this section presents research findings in detail, regarding the investigation of the original research questions formulated in Section 1.2. We list the research questions below and provides research findings regarding each of them.

Research Question 1. *How to design suitable deep neural network models for processing sensor and social media data in smart city?*

This question is regarding the model selection for different types of city data. We have shown the data representation of sensor, image or video, and text data representation in Section 3.2, and discussed the model selection for different modalities in Section 2.3.6. In general, RNN is the first choice for modelling sequence data with strong temporal dependencies such as sensor data; BERT outperforms other popular models, e.g., CNN and RNN, in terms of processing textual data such as social media messages.

Sensor data: On the basis of RNN, we developed two deep learning models in Section

4.1 and 5.1 to processing traffic flow data. In Section 4.1, an attention recurrent neural network is used to extract knowledge from historical traffic flow reading with different importance for prediction; the encoder-decoder architecture of RNN is utilised to generate a sequences of prediction over the next few hours. In Section 5.1, a Sensor decoder with two LSTM-RNN layers are used to extract traffic event information, which outperforms the CNN model as shown in Table 5.2.

Social media data: We have conducted extensively experiments for processing social media data with different deep learning models in Section 6. Table 6.2 showed that the three baseline models (i.e. LSTM, Bi-LSTM, CNN) can detect and classify tweets successfully within a certain degree of effectiveness. As the glove input vectors to these three models are the same, the prediction accuracies of these three models are similar. When we compare these context-free models with the BERT model, we observe significant improvements in prediction accuracy for all cases ranging from 0.83% to 6.04%. This observation demonstrates the effectiveness of fine-tuning a large pre-trained contextual model for social media data processing.

Technically, it is difficult to make a straightforward conclusion on what the best model is for a particular problem. For example, sensor data representations can be transformed into 2D matrix, and processed with CNN [184]; BERT may take significantly more running time compared with models directly using word embeddings (as shown in Table 6.6), which may not suitable for capability-constrained IoT devices. We expect to see more deep learning models will be developed by the research community for different purposes in smart city applications.

Research Question 2. *How to design a new model to perform cross-domain knowledge discovery?*

Smart city is a Cyber-Physical-Social System, where data comes from cyber world, physical world and social world. Data from different sources with different modalities can complement each other and more comprehensive knowledge could be derived. In Chapter 5, we present two applications of multi-modal data processing. In Section 5.1, we designed a sensor encoder and a social encoder for traffic flow data and social media data respectively. Multi-modal Network (MMN) is used to concatenate features from different modalities for prediction. Similarly in Section 5.2, a MMN is utilised to extract knowledge from image and text data for crisis-relate data classification task. As reported in Table 5.2 and 5.3,

multi-modal models outperform all single-modal models (i.e., RNN, CNN) in accuracy ranging from 0.97% to 28.48%. It should be noted that advanced fusion methods, e.g., using attention mechanism, may better filter uninformative and misleading components from cross-domain data and result in better performance.

Research Question 3. *How to address deep learning limitations in real-world smart city applications?*

Different form benchmark datasets in CV and NLP tasks, the real-world applications contains a number of challenges to be addressed. In this thesis, we tried to addressed these issues summarised as follows.

- **external knowledge integration:** In real-world scenarios, result may affected by a number of external factors. A general fusion component could be used to transfer these external information into numeric vectors and integrated into the model for better performance as described in Section 4.1.

- **model robustness:** DNN models are vulnerable to adversarial examples. Small perturbations to the input will cause the DNN models to generate incorrect results. Adversarial training technique is applied to improve the robustness of the deep neural network against adversarial examples for crisis-related tweet classification in Section 5.2.

- **semi-supervised learning:** Acquiring large amount of labelled data for model training in real-world scenario is impractical. We extend GAN to a semi-supervised learning framework for traffic event detection in Section 5.1, which use the rich amount of unlabelled data and the extremely limited amount of labelled one.

- **unsupervised learning:** We further explore the ability of learning from unlabelled data. In Chapter 6, we applied domain adaptation and zero-shot learning techniques to crisis data classification and COVID-19 tweet classification tasks respectively, and achieved promising results.

7.3 Future Work

Previously, we have summarised some limitations and directions of deep learning and smart city separately in Section 2.4 and Section 3.4. In this section, we will discuss several future work directions for deep learning from smart city data. These directions are summarised as follows.

Emerging Techniques: Various novel deep learning techniques and architectures are proposed every year. It seems that the researchers from the smart city domain have not been fully aware of some latest development in deep learning. Some networks may have great potential in smart city applications. For example, the original design of Generative Adversarial Network (GAN) is for image generation purpose, while the network can be extended to address the semi-supervised issue for a traffic event detection task as described in Section 5.1. Graph Neural Network (GNN) is a class of deep learning method designed to perform inference on data described by graphs. One typical application of GNN in smart city is traffic prediction, where the traffic network can be modeled as a spatial-temporal graph; the nodes are loop sensors installed on roads, and the edges represent the intersections or road segments connecting these sensors. Leveraging these state-of-the-art deep learning techniques in properly solving smart city tasks is a interesting direction.

Distributed Intelligence: The extraordinarily large amount of heterogeneous data (e.g., numerical, image, text) collected from different sources (e.g., sensors networks, hospitals, and social media) for cities is a type of big data. Moving this big volume of data from the network edge to a central data center for processing and analysis not only adds latency but also consumes network bandwidth. Therefore, the cloud-based IoT with a centralized data center may not be able to enable smart environments, such as automated vehicles, traffic controls, etc., whose data need to be analysed and acted on quickly. According to the idea of "distributed intelligence", data processing and analysis should be performed where the data are collected to tackle the challenges brought by the volume and velocity of big city data. Thus, we have to take into consideration the capability of these computing devices as they are far less powerful compared to servers in data centres. Currently, a lot of research studies built extremely deep and complex networks, which may take days or even weeks to train, to get the ultimate accuracy. It is impractical that these models can be implemented on the capability-constrained IoT devices for real-time prediction. Therefore, how to design more lightweight models (can be used on IoT devices) with competitive performance would be an interesting future direction.

Social IoT: 'human as sensors' or 'citizen sensing' has become a popular phenomenon for which humans are not only the data users, but also the data providers. Enormous amount of social media data can be collected and further processed and analyzed in various downstream tasks which may have great influence on human society. Such large amount of social media data covers nearly everything happening around the world has become valuable resources for various disciplines. Algorithm a number of machine learning models, e.g., SVM, RF, RNN, CNN, BERT, etc., have been applied to extract knowledge from social media, it is still not well-investigated. On the one hand, Social media data is usually noisy, dynamic and contains many abbreviations that cannot be recognised by NLP techiniques such as word2vec or BERT. A possible direction is to combine the current NLP models with external knowledge bases or knowledge graphs, which contain enough common sense knowledge, for better natural language understanding. On the other hand, streaming social media data may have spatial and temporal relations among them, while such relations are usually ignored by current studies that only process messages separately. Streaming and batch processing methods with deep learning models should be considered to better capture features or events from social media data.

Cross-domain Knowledge Discovery: So far, smart city has appeared as a collection of loosely coupled applications in different domains, and little relation can be identified from these applications. The current research is much like applying deep learning to identify and extract superficial knowledge. Data from different domains may be correlated and may affect each other, e.g., bad weather may affect traffic conditions, traffic congestion may result in extra fuel consumption, air and noise pollution, and such pollution may cause cardiovascular and respiratory disease. However, such correlations have been largely overlooked in the existing research. Cross-domain knowledge discovery would be an essential component to uncover the concealed insight beneath the big city data from different domains, and eventually help with building truly smart cities.

Theoretical Analysis: Although deep learning has achieved promising results in a number of research projects, there is still a lack of theoretical analysis in many studies, especially those related to deep learning applications. The validity of the proposed models in these studies is usually only examined by performances on benchmark datasets. It remains a problem because it is unclear whether 1 or 2% is statistically an improvement or whether the results may have occurred by changing the parameters. More theoretical analysis should be included in the studies to better show why the proposed model is effective, under what conditions and how to apply to different studies. Theoretical analysis

will be one of the major concerns of our future work.

7.4 Epilogue

The work reported in this thesis focuses on exploring deep learning methods for smart city applications. The thesis provides a review of the latest research on the convergence of deep learning and smart city for data processing from a technique-oriented perspective and an application-oriented perspective. We designed a number of deep neural network with respect to improving and extending the current studies in smart city area. We focused on two smart city area (intelligence transportation and social media analysis) with three types of data modalities (sensor, social media and multi-modal data). In addition, various components or techniques are integrated into the proposed models to address challenges, i.e., external knowledge integration, model robustness, multi-modal knowledge fusion, semi-supervised and unsupervised learning, and have achieved superior performance over the existing baselines. We have also discussed a number of future directions of deep learning from smart city data. The work presented in this thesis would be beneficial to further research in the area of deep learning and smart city, and hopefully would move the relevant research on step further.

Appendix A

Publications

- Q. Chen, W. Wang, K. Huang and F. Coenen . Zero-shot Text Classification via Knowledge Graph Embedding for Social Media Data. *IEEE Internet of Things Journal*, 2021, Accepted.

- Q. Chen, W. Wang, K. Huang, S. De and F. Coenen . Multi-modal Generative Adversarial Networks for Traffic Event Detection in Smart Cities. *Expert Systems with Applications*, 2021.

- Q. Chen, W. Wang, K. Huang, S. De and F. Coenen . Adversarial Domain Adaptation for Crisis Data Classification on Social Media. *International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, pp.282-287, 2020.

- Q. Chen, W. Wang, K. Huang, S. De and F. Coenen . Multi-modal Adversarial Training for Crisis-related Data Classification on Social Media. *IEEE International Conference on Smart Computing (SMARTCOMP)*, pp.232-237, 2020.

- Q. Chen, W. Wang, X. Huang, and H. N. Liang. Attention-based Recurrent Neural Network for Traffic Flow Prediction. *Journal of Internet Technology.* 21(3): 831-839, 2020.

- Q. Chen, W. Wang, F. Wu, S. De, R. Wang, B. Zhang, and X. Huang. A survey on an

emerging area: Deep learning for smart city data, *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(5), 392-410, 2019.

- Q. Chen and W. Wang. Multi-modal Neural Network for Traffic Event Detection. *IEEE 2nd International Conference on Electronics and Communication Engineering (ICECE)*, pp.26-30, 2019.

- Q. Chen, W. Wang, and X. Huang. Long Short-Term Memory Encoder-Decoder for Traffic Flow Prediction. *International Conference on Recent Advancements in Computing, IoT and Computer Engineering Technology*, 2018.

Bibliography

[1] Mahdi Abavisani, Liwei Wu, Shengli Hu, Joel Tetreault, and Alejandro Jaimes. Multimodal categorization of crisis events in social media. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14679–14689, 2020.

[2] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.

[3] Jaehyun Ahn, Dongil Shin, Kyuho Kim, and Jihoon Yang. Indoor air quality analysis using deep learning with sensor data. *Sensors*, 17(11):2476, 2017.

[4] Qeethara Kadhim Al-Shayea. Artificial neural networks in medical diagnosis. *International Journal of Computer Science Issues*, 8(2):150–154, 2011.

[5] Firoj Alam, Shafiq Joty, and Muhammad Imran. Domain adaptation with adversarial training and graph embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1087, 2018.

[6] Firoj Alam, Ferda Ofli, and Muhammad Imran. Crisismmd: Multimodal twitter datasets from natural disasters. In *Twelfth International AAAI Conference on Web and Social Media*, pages 465–473, 2018.

[7] Signorini Alessio, Segre Alberto Maria, and Philip M Polgreen. The use of twitter to track levels of disease activity and public concern in the U.S. during the influenza a H1N1 pandemic. *Plos One*, 6(5):e19467, 2011.

[8] Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*, 2019.

[9] Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Carlo Meghini, and Claudio Vairo. Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, 72:327–334, 2017.

[10] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

[11] Christos-Nikolaos E Anagnostopoulos, Ioannis E Anagnostopoulos, Ioannis D Psoroulas, Vassili Loumos, and Eleftherios Kayafas. License plate recognition from still images and video sequences: A survey. *IEEE Transactions on intelligent transportation systems*, 9(3):377–391, 2008.

[12] Pramod Anantharam, Payam Barnaghi, Krishnaprasad Thirunarayan, and Amit Sheth. Extracting city traffic events from social streams. *ACM Transactions on Intelligent Systems and Technology*, 6(4):43, 2015.

[13] Pramod Anantharam, Krishnaprasad Thirunarayan, Surendra Marupudi, Amit Sheth, and Tanvi Banerjee. Understanding city traffic dynamics utilizing sensor and textual observations. In *Thirtieth AAAI Conference on Artificial Intelligence*, pages 3793–3799, 2016.

[14] Marios Anthimopoulos, Stergios Christodoulidis, Lukas Ebner, Andreas Christe, and Stavroula Mougiakakou. Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE transactions on medical imaging*, 35(5):1207–1216, 2016.

[15] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[16] MR Avendi, Arash Kheradvar, and Hamid Jafarkhani. A combined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac MRI. *Medical image analysis*, 30:108–119, 2016.

[17] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[18] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

[19] Payam Barnaghi, Wei Wang, Lijun Dong, and Chonggang Wang. A linked-data model for semantic sensor streams. In *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, pages 468–475, 2013.

[20] Brett K Beaulieu-Jones and Jason H Moore. Missing data imputation in the electronic health record using deeply learned autoencoders. In *Pacific Symposium on Biocomputing 2017*, pages 207–218, 2017.

[21] Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. In *Semi-Supervised Learning*, 2006.

[22] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[23] Caltrans. Performance measurement system (pems), 2019. [Online].

[24] Zhimin Cao, Qi Yin, Xiaoou Tang, and Jian Sun. Face recognition with learning-based descriptor. In *IEEE Computer society conference on computer vision and pattern recognition*, pages 2707–2714. IEEE, 2010.

[25] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

[26] Kit Yan Chan, Tharam S Dillon, Jaipal Singh, and Elizabeth Chang. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and levenberg–marquardt algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):644–654, 2012.

[27] H Chang, Youngjoo Lee, B Yoon, and Sanghoon Baek. Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences. *IET intelligent transport systems*, 6(3):292–305, 2012.

[28] Dong Chen, Xudong Cao, Fang Wen, and Jian Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3025–3032, 2013.

[29] Yuanyuan Chen, Yisheng Lv, and Fei-Yue Wang. Traffic flow imputation using parallel data and generative adversarial networks. *IEEE Transactions on Intelligent Transportation Systems*, 21(4):1624–1630, 2019.

[30] Yu-Chiun Chiou, Lawrence W Lan, Chun-Ming Tseng, and Chih-Chao Fan. Optimal locations of license plate recognition to enhance the origin-destination matrix estimation. In *Proceedings of the Eastern Asia Society for Transportation Studies The 9th International Conference of Eastern Asia Society for Transportation Studies*, pages 297–297, 2011.

[31] Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[32] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. Generating multi-label discrete patient records using generative adversarial networks. In *Machine Learning for Healthcare Conference*, pages 286–305, 2017.

[33] Amsterdam Smart City. https://amsterdamsmartcity.com/. accessed: 2020-08-12.

[34] Michael Compton, Payam Barnaghi, Luis Bermudez, Ral Garca Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. The SSN Ontology of the Semantic Sensor Networks Incubator Group. *Journal of Web Semantics*, 17:2–32, 2012.

[35] Andrew Crooks, Arie Croitoru, Anthony Stefanidis, and Jacek Radzikowski. Earthquake: Twitter as a distributed sensor system. *Transactions in Gis*, 17(1):124–147, 2013.

[36] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4883–4894, 2019.

[37] Sina Dabiri and Kevin Heaslip. Developing a twitter-based traffic event detection model using deep learning architectures. *Expert Systems with Applications*, 118:425–439, 2019.

[38] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pages 6510–6520, 2017.

[39] Eleonora D'Andrea, Pietro Ducange, Beatrice Lazzerini, and Francesco Marcelloni. Real-time detection of traffic from twitter stream analysis. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):2269–2283, 2015.

[40] Eleonora D'Andrea and Francesco Marcelloni. Detection of traffic congestion and incidents from gps trace analysis. *Expert Systems with Applications*, 73:43–56, 2017.

[41] Suparna De, Yuchao Zhou, Iker Larizgoitia Abad, and Klaus Moessner. Cyber-physical-social frameworks for urban big data systems: A survey. *Applied Sciences*, 7(10):1017, 2017.

[42] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[43] Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, 2015.

[44] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

[45] Yanjie Duan, Yisheng Lv, Yu-Liang Liu, and Fei-Yue Wang. An efficient realization of deep learning for traffic data imputation. *Transportation research part C: emerging technologies*, 72:168–181, 2016.

[46] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[47] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, 2015.

[48] Rui Fu, Zuo Zhang, and Li Li. Using LSTM and GRU neural network methods for traffic flow prediction. In *31st Youth Academic Annual Conference of Chinese Association of Automation*, pages 324–328, 2016.

[49] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189, 2015.

[50] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, volume 70, pages 1243–1252, 2017.

[51] Deepanway Ghosal, Devamanyu Hazarika, Navonil Majumder, Abhinaba Roy, Soujanya Poria, and Rada Mihalcea. Kingdom: Knowledge-guided domain adaptation for sentiment analysis. *arXiv preprint arXiv:2005.00791*, 2020.

[52] Fréderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle. Multimedia lab@ acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations. In *Proceedings of the workshop on noisy user-generated text*, pages 146–153, 2015.

[53] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[54] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[55] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[56] Manu Goyal, Neil D Reeves, Adrian K Davison, Satyan Rajbhandari, Jennifer Spragg, and Moi Hoon Yap. Dfunet: Convolutional neural networks for diabetic foot ulcer classification. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(5):728–739, 2018.

[57] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649, 2013.

[58] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[59] Yiming Gu, Zhen Qian, and Feng Chen. From twitter to detector: real-time traffic incident detection using social media data. *Transportation Research Part C*, 67:321–342, 2016.

[60] Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. Smart grid technologies: Communication technologies and standards. *IEEE transactions on Industrial informatics*, 7(4):529–539, 2011.

[61] Hongliang Guo, Zhiguang Cao, Madhavan Seshadri, Jie Zhang, Dusit Niyato, and Ulrich Fastenrath. Routing multiple vehicles cooperatively: Minimizing road network breakdown probability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(2):112–124, 2017.

[62] Zhiwei Guo, Yu Shen, Ali Kashif Bashir, Muhammad Imran, Neeraj Kumar, Di Zhang, and Keping Yu. Robust spammer detection using collaborative neural network in internet-of-things applications. *IEEE Internet of Things Journal*, 8(12):9549–9558, 2020.

[63] Nils Yannick Hammerla, James Fisher, Peter Andras, Lynn Rochester, Richard Walker, and Thomas Plötz. PD disease state assessment in naturalistic environments using deep learning. In *Twenty-Ninth AAAI conference on artificial intelligence*, pages 1742–1748, 2015.

[64] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[65] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[66] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[67] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[68] Shin Hoo-Chang, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285, 2016.

[69] Jen-Cheng Hou, Syu-Siang Wang, Ying-Hui Lai, Yu Tsao, Hsiu-Wen Chang, and Hsin-Min Wang. Audio-visual speech enhancement using multimodal deep convolutional neural networks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):117–128, 2018.

[70] J Brian Houston, Joshua Hawthorne, Mildred F Perreault, Eun Hae Park, Marlo Goldstein Hode, Michael R Halliwell, Sarah E Turner McGowen, Rachel Davis, Shivani Vaid, Jonathan A McElderry, et al. Social media and disasters: a functional framework for social media use in disaster planning, response, and research. *Disasters*, 39(1):1–22, 2015.

[71] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[72] Chang Di Huang, Chien Yao Wang, and Jia Ching Wang. Human action recognition system for elderly and children care using three stream ConvNet. In *International Conference on Orange Technologies*, pages 5–9, 2016.

[73] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[74] Wenhao Huang, Guojie Song, Haikun Hong, and Kunqing Xie. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):2191–2201, 2014.

[75] Muhammad Imran, Shady Elbassuoni, Carlos Castillo, Fernando Diaz, and Patrick Meier. Practical extraction of disaster-relevant information from social media. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1021–1024, 2013.

[76] Muhammad Imran, Prasenjit Mitra, and Carlos Castillo. Twitter as a lifeline: Human-annotated twitter corpora for nlp of crisis-related messages. *arXiv preprint arXiv:1605.05894*, 2016.

[77] Young-Seon Jeong, Young-Ji Byon, Manoel Mendonca Castro-Neto, and Said M Easa. Supervised weighting-online learning algorithm for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1700–1707, 2013.

[78] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

[79] Milton keynes smart city. http://www.mksmart.org/. accessed: 2020-08-12.

[80] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[81] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[82] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[83] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

[84] Rob Kitchin. The real-time city? big data and smart urbanism. *GeoJournal*, 79(1):1–14, 2014.

[85] Arief Koesdwiady, Ridha Soua, and Fakhreddine Karray. Improving traffic flow prediction with weather information in connected cars: a deep learning approach. *IEEE Transactions on Vehicular Technology*, 65(12):9508–9517, 2016.

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

[87] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[88] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[89] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195, 2015.

[90] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[91] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3911–3919, 2017.

[92] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, and Yu Zheng. Geoman: Multi-level attention networks for geo-sensory time series prediction. In *International Joint Conference on Artificial Intelligence*, volume 2018, pages 3428–3434, 2018.

[93] Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzel. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, 2015.

[94] G Litjens, T Kooi, B. E. Bejnordi, Setio Aaa, F Ciompi, M Ghafoorian, Van Der Laak Jawm, Ginneken B Van, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42(9):60–88, 2017.

[95] Guixiang Liu, Zhongyou Ma, Zhongguo Du, and Can Wen. The calculation method of road travel time based on license plate recognition technology. In *Advances in information technology and education*, pages 385–389. Springer, 2011.

[96] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2901–2908, 2020.

[97] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *European Conference on Computer Vision*, pages 869–884, 2016.

[98] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[99] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[100] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. Traffic flow prediction with big data: a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865–873, 2015.

[101] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54:187–197, 2015.

[102] David Menotti, Giovani Chiachia, Alexandre X Falcão, and VJ Oliveira Neto. Vehicle license plate recognition with random convolutional networks. In *27th SIBGRAPI conference on graphics, patterns and images*, pages 298–303, 2014.

[103] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv: Computation and Language*, 2013.

[104] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[105] Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6:26094, 2016.

[106] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4207–4215, 2016.

[107] Naser Hossein Motlagh, Miloud Bagaa, and Tarik Taleb. UAV-based IoT platform: a crowd surveillance use case. *IEEE Communications Magazine*, 55(2):128–134, 2017.

[108] Naser Hossein Motlagh, Tarik Taleb, and Osama Arouk. Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives. *IEEE Internet of Things Journal*, 3(6):899–922, 2016.

[109] Khan Muhammad, Jamil Ahmad, and Sung Wook Baik. Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Neurocomputing*, 288:30–42, 2018.

[110] Abdulmajid Murad and Jae-Young Pyun. Deep recurrent neural networks for human activity recognition. *Sensors*, 17(11):2556, 2017.

[111] Meenakshi Nagarajan, Karthik Gomadam, Amit P Sheth, Ajith Ranabahu, Raghava Mutharaju, and Ashutosh Jadhav. Spatio-temporal-thematic analysis of citizen sensor data: Challenges and experiences. In *International Conference on Web Information Systems Engineering*, pages 539–553, 2009.

[112] Loris Nanni, Alessandra Lumini, and Sheryl Brahnam. Local binary patterns variants as texture descriptors for medical image analysis. *Artificial intelligence in medicine*, 49(2):117–125, 2010.

[113] United Nations. World Urbanization Prospects The 2018 Revision. Technical report, Department of Economic and Social Affairs, United Nations, 2018.

[114] Dat T Nguyen, Ferda Ofli, Muhammad Imran, and Prasenjit Mitra. Damage assessment from social media imagery data during disasters. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 569–576, 2017.

[115] Dat Tien Nguyen, Kamela Ali Al Mannai, Shafiq Joty, Hassan Sajjad, Muhammad Imran, and Prasenjit Mitra. Robust classification of crisis-related data on social networks using convolutional neural networks. In *Eleventh International AAAI Conference on Web and Social Media*, pages 632–635, 2017.

[116] Dat Tien Nguyen, Kamela Ali Al Mannai, Shafiq Joty, Hassan Sajjad, Muhammad Imran, and Prasenjit Mitra. Rapid classification of crisis-related data on social networks using convolutional neural networks. *arXiv preprint arXiv:1608.03902*, 2016.

[117] National Oceanic and Atmospheric Administration (NOAA). State of the science fact sheet. Technical report, United States Department of Commerce, 2016.

[118] Christopher Olah. Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/. accessed: 2018-09-14.

[119] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo. What to expect when the unexpected happens: Social media communications across crises. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 994–1009, 2015.

[120] Bun Theang Ong, Komei Sugiura, and Koji Zettsu. Dynamically pre-trained deep recurrent neural networks using environmental monitoring data for predicting PM2. 5. *Neural Computing and Applications*, 27(6):1553–1566, 2016.

[121] Catherine Ordun, Sanjay Purushotham, and Edward Raff. Exploratory analysis of covid-19 tweets using topic modeling, umap, and digraphs. *arXiv preprint arXiv:2005.03082*, 2020.

[122] Antonio M Ortiz, Dina Hussein, Soochang Park, Son N Han, and Noel Crespi. The cluster between internet of things and social networks: Review and research challenges. *IEEE Internet of Things Journal*, 1(3):206–215, 2014.

[123] Omaima FathElrahman Osman and Moi Hoon Yap. Computational intelligence in automatic face age estimation: A survey. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(99):1–15, 2018.

[124] Bei Pan, Yu Zheng, David Wilkie, and Cyrus Shahabi. Crowd sensing of traffic anomalies based on human mobility and social media. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 344–353, 2013.

[125] Gang Pan, Guande Qi, Wangsheng Zhang, Shijian Li, Zhaohui Wu, and Laurence Tianruo Yang. Trace analysis and mining for smart cities: issues, methods, and applications. *IEEE Communications Magazine*, 51(6):120–126, 2013.

[126] Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016-2016 IEEE Military Communications Conference*, pages 49–54, 2016.

[127] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pages 1532–1543, 2014.

[128] Alex Pentland and Tanzeem Choudhury. Face recognition for smart environments. *Computer*, 33(2):50–55, 2000.

[129] Joao Pereira, Arian Pasquali, Pedro Saleiro, and Rosaldo Rossetti. Transportation in social media: an automatic classifier for travel-related tweets. In *EPIA Conference on Artificial Intelligence*, pages 355–366, 2017.

[130] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[131] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[132] Parisa Pouladzadeh, Pallavi Kuhad, Sri Vijay Bharat Peddi, Abdulsalam Yassine, and Shervin Shirmohammadi. Food calorie measurement using deep learning neural network. In *IEEE International Instrumentation and Measurement Technology Conference Proceedings*, pages 1–6, 2016.

[133] Smartsantander project. http://www.smartsantander.eu/. accessed: 2020-08-12.

[134] Pushpankar Kumar Pushp and Muktabh Mayank Srivastava. Train once, test anywhere: Zero-shot learning for text classification. *arXiv preprint arXiv:1712.05972*, 2017.

[135] Umair Qazi, Muhammad Imran, and Ferda Ofli. Geocov19: a dataset of hundreds of millions of multilingual covid-19 tweets with location information. *SIGSPATIAL Special*, 12(1):6–15, 2020.

[136] Zhongang Qi, Tianchun Wang, Guojie Song, Weisong Hu, Xi Li, and Zhongfei Mark Zhang. Deep air learning: Interpolation, prediction, and feature analysis of fine-grained air quality. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[137] Yongrui Qin, Quan Z. Sheng, Nickolas J.G. Falkner, Schahram Dustdar, Hua Wang, and Athanasios V. Vasilakos. When things matter: A survey on data-centric internet of things. *Journal of Network and Computer Applications*, 64:137 – 153, 2016.

[138] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018.

[139] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[140] Elad Richardson, Matan Sela, Roy Or-El, and Ron Kimmel. Learning detailed face reconstruction from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5553–5562, 2017.

[141] Herbert Robbins and Sutton Monro. A stochastic approximation method. In *Herbert Robbins Selected Papers*, pages 102–109. Springer, 1985.

[142] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015.

[143] Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, 2019.

[144] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3379–3388, 2018.

[145] Anuradha Saha, Amit Konar, and Atulya K Nagar. EEG Analysis for Cognitive Failure Detection in Driving Using Type-2 Fuzzy Classifiers. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(6):437–453, 2017.

[146] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[147] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. Interpretable adversarial perturbation in input embedding space for text. *arXiv preprint arXiv:1805.02917*, 2018.

[148] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[149] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[150] Aliaksei Severyn and Alessandro Moschitti. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 959–962, 2015.

[151] Amit Sheth. Citizen sensing, social signals, and enriching human experience. *IEEE Internet Computing*, 13(4):87–92, 2009.

[152] Palaiahnakote Shivakumara, Dongqi Tang, Maryam Asadzadehkaljahi, Tong Lu, Umapada Pal, and Mohammad Hossein Anisi. Cnn-rnn based method for license plate recognition. *CAAI Transactions on Intelligence Technology*, 3(3):169–175, 2018.

[153] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[154] Barcelona smart city. http://smartcity.bcn.cat/en/projects. accessed: 2020-08-12.

[155] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*, 2019.

[156] Jakub Sochor, Adam Herout, and Jiri Havel. Boxcars: 3d boxes as CNN input for improved fine-grained vehicle recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3006–3015, 2016.

[157] Moataz Soliman, Tobi Abiodun, Tarek Hamouda, Jiehan Zhou, and Chung-Horng Lung. Smart home: Integrating internet of things with web services and cloud computing. In *IEEE 5th international conference on cloud computing technology and science*, volume 2, pages 317–320, 2013.

[158] Xuan Song, Hiroshi Kanasugi, and Ryosuke Shibasaki. DeepTransport: prediction and simulation of human mobility and transportation mode at a citywide level. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2618–2624, 2016.

[159] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4444–4451, 2017.

[160] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[161] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[162] Masanori Suganuma, Shinichi Shirakawa, and Tomoharu Nagao. A genetic programming approach to designing convolutional neural network architectures. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 497–504, 2017.

[163] Yanan Sun, Gary G Yen, and Zhang Yi. Evolving unsupervised deep neural networks for learning meaningful representations. *IEEE Transactions on Evolutionary Computation*, 23(1):89–103, 2019.

[164] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1891–1898, 2014.

[165] Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 14–25, 2014.

[166] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

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

[168] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[169] Yong Tang, Congzhe Zhang, Renshu Gu, Peng Li, and Bin Yang. Vehicle detection and recognition for intelligent traffic surveillance system. *Multimedia tools and applications*, 76(4):5817–5832, 2017.

[170] Yongxue Tian and Li Pan. Predicting short-term traffic flow by long short-term memory recurrent neural network. In *IEEE International Conference on Smart City/socialcom/sustaincom*, pages 153–158, 2015.

[171] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[172] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1415–1424, 2017.

[173] Kumar Utkarsh, Anupam Trivedi, Dipti Srinivasan, and Thomas Reindl. A consensus-based distributed computational intelligence technique for real-time optimal control in smart distribution grids. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(1):51–60, 2017.

[174] Sepehr Valipour, Mennatullah Siam, Eleni Stroulia, and Martin Jagersand. Parking-stall vacancy indicator system, based on deep convolutional neural networks. In *Internet of Things*, pages 655–660, 2016.

[175] Lelitha Vanajakshi and Laurence R Rilett. Support vector machine technique for the short term prediction of travel time. In *IEEE Intelligent Vehicles Symposium*, pages 600–605, 2007.

[176] Upkar Varshney. Pervasive healthcare and wireless health monitoring. *Mobile Networks and Applications*, 12(2-3):113–127, 2007.

[177] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[178] Anand Vetrivel, Markus Gerke, Norman Kerle, Francesco Nex, and George 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*, 140:45–59, 2018.

[179] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[180] Dimitris Voukantsis, Kostas Karatzas, Jaakko Kukkonen, Teemu Räsänen, Ari Karppinen, and Mikko Kolehmainen. Intercomparison of air quality data using principal component analysis, and forecasting of pm10 and pm2. 5 concentrations using artificial neural networks, in thessaloniki and helsinki. *Science of the Total Environment*, 409(7):1266–1276, 2011.

[181] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234, 2016.

[182] Di Wang, Ahmad Al-Rubaie, Sandra Stinčić Clarke, and John Davies. Real-time traffic event detection from social media. *ACM Transactions on Internet Technology*, 18(1):9, 2017.

[183] Dong Wang, Junbo Zhang, Wei Cao, Jian Li, and Yu Zheng. When Will You Arrive? Estimating Travel Time Based on Deep Neural Networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[184] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, 119:3–11, 2019.

[185] Puming Wang, Laurence T Yang, Jintao Li, Jinjun Chen, and Shangqing Hu. Data fusion in cyber-physical-social systems: State-of-the-art and perspectives. *Information Fusion*, 51:42–57, 2019.

[186] W. Wang, S. De, Y. Zhou, X. Huang, and K. Moessner. Distributed sensor data computing in smart city applications. In *IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks*, pages 1–5, 2017.

[187] Wei Wang and Min Zhang. Tensor deep learning model for heterogeneous data fusion in internet of things. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(1):32–41, 2018.

[188] Xiaoyu Wang, Tony X Han, and Shuicheng Yan. An HOG-LBP human detector with partial occlusion handling. In *IEEE 12th international conference on computer vision*, pages 32–39, 2009.

[189] Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 849–857, 2018.

[190] Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

[191] Qi Wu, Chingchun Huang, Shih-yu Wang, Wei-chen Chiu, and Tsuhan Chen. Robust parking space detection considering inter-space correlation. In *IEEE International Conference on Multimedia and Expo*, pages 659–662, 2007.

[192] Xiang Wu, Ran He, and Zhenan Sun. A lightened cnn for deep face representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 4, page 5, 2015.

[193] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[194] Jun Xu, Rouhollah Rahmatizadeh, Ladislau Bölöni, and Damla Turgut. Real-time prediction of taxi demand using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 19(8):2572–2581, 2017.

[195] Hao-Fan Yang, Tharam S Dillon, and Yi-Ping Phoebe Chen. Optimized structure of the traffic flow forecasting model with a deep learning approach. *IEEE transactions on neural networks and learning systems*, 28(10):2371–2381, 2017.

[196] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3973–3981, 2015.

[197] Xiuwen Yi, Junbo Zhang, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. Deep distributed fusion network for air quality prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 965–973, 2018.

[198] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. *arXiv preprint arXiv:1909.00161*, 2019.

[199] Kevin Y Yip, Chao Cheng, and Mark Gerstein. Machine learning and genome annotation: a match meant to be? *Genome biology*, 14(5):1–10, 2013.

[200] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32, 2014.

[201] Shlomo Zeitman. Parking management system, August 17 1999. US Patent 5,940,481.

[202] Jing Zeng, Laurence T Yang, Man Lin, Huansheng Ning, and Jianhua Ma. A survey: Cyber-physical-social systems and their system-level design methodology. *Future Generation Computer Systems*, 105:1028–1042, 2020.

[203] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-first AAAI conference on artificial intelligence*, pages 1655–1661, 2017.

[204] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.

[205] Nan Zhang, Fei Yue Wang, Fenghua Zhu, Dongbin Zhao, and Shuming Tang. DynaCAS: Computational Experiments and Decision Support for ITS. *IEEE Intelligent Systems*, 23(6):19–23, 2008.

[206] Richong Zhang, Fanshuang Kong, Chenyue Wang, and Yongyi Mao. Embedding of hierarchically typed knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2046–2053, 2018.

[207] Shanghang Zhang, Guanhang Wu, Joao P Costeira, and Jose MF Moura. Understanding traffic density from large-scale web camera data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5898–5907, 2017.

[208] Wangsheng Zhang, Guande Qi, Gang Pan, Hua Lu, Shijian Li, and Zhaohui Wu. City-scale social event detection and evaluation with taxi traces. *ACM Transactions on Intelligent Systems and Technology*, 6(3):40, 2015.

[209] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, 2019.

[210] Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via semantic similarity embedding. In *Proceedings of the IEEE international conference on computer vision*, pages 4166–4174, 2015.

[211] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology*, 5(3):38, 2014.

[212] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3754–3762, 2017.

[213] Fan Zhou, Qing Yang, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ashfaq Khokhar. Reinforced spatiotemporal attentive graph neural networks for traffic forecasting. *IEEE Internet of Things Journal*, 7(7):6414–6428, 2020.

[214] Y. Zhou, S. De, W. Wang, and K. Moessner. Enabling query of frequently updated data from mobile sensing sources. In *IEEE 17th International Conference on Computational Science and Engineering*, pages 946–952, 2014.

[215] Yao Zhou, Hua Mao, and Zhang Yi. Cell mitosis detection using deep neural networks. *Knowledge-Based Systems*, 137:19–28, 2017.

[216] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.