2018-07-31

Deep Learning Based Imbalanced Data Classification and Information Retrieval for Multimedia Big Data

Yilin Yan
University of Miami, y.yan4@umiami.edu

UNIVERSITY OF MIAMI

DEEP LEARNING BASED IMBALANCED DATA CLASSIFICATION AND INFORMATION RETRIEVAL FOR MULTIMEDIA BIG DATA

By

Yilin Yan

A DISSERTATION

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Coral Gables, Florida

August 2018

UNIVERSITY OF MIAMI

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

DEEP LEARNING BASED IMBALANCED DATA CLASSIFICATION AND INFORMATION RETRIEVAL FOR MULTIMEDIA BIG DATA

Yilin Yan

Approved:

Mei-Ling Shyu, Ph.D.
Professor of Electrical and
Computer Engineering

Onur Tigli, Ph.D.
Associate Professor of Electrical and
Computer Engineering

Jie Xu, Ph.D.
Assistant Professor of Electrical and
Computer Engineering

Hammam Alsafrjalani, Ph.D.
Assistant Professor in Practice of
Electrical and Computer Engineering

Shu-Ching Chen, Ph.D.
Professor of School of Computing
and Information Sciences
Florida International University

Guillermo Prado, Ph.D.
Dean of the Graduate School

YAN, YILIN (Ph.D., Electrical and Computer Engineering)

Deep Learning Based Imbalanced Data Classification and Information Retrieval for Multimedia Big Data (August 2018)

Abstract of a dissertation at the University of Miami.

Dissertation supervised by Professor Mei-Ling Shyu.
No. of pages in text. (153)

The development in information science has enabled an explosive growth of data, which attracts more and more researchers to engage in the field of big data analytics. Noticeably, in many real-world applications, large amounts of data are imbalanced data since the events of interests occur infrequently. Classification of imbalanced data is an important research problem as lots of real-world datasets have skewed class distributions in which the majority of instances (examples) belong to one class and far fewer instances belong to the others. A classifier induced from an imbalanced dataset is more likely to be biased towards the majority classes and shows very poor classification accuracy on the minority classes. While in many applications, the minority instances actually represent the concept of interest (e.g., fraud in banking operations, abnormal cell in medical data, etc.), and the detection of these rare events has become more important. Despite extensive research efforts, rare event mining remains one of the most challenging problems in information retrieval, especially for multimedia big data.

To tackle this challenge, in this dissertation, we propose an extended deep learning approach to achieve promising performance in classifying largely skewed multimedia dataset. Specifically, we investigate the integration of bootstrapping methods and a state-of-the-art deep learning approach, Convolutional Neural Networks (CNNs),

with extensive empirical studies. Considering the fact that deep learning approaches such as CNNs are usually computationally expensive, we propose to feed low-level features to CNNs and prove its feasibility in achieving promising performance while saving a lot of training time. Furthermore, since big training datasets are required to train CNNs, we propose to extract features from pre-trained CNN models and feed those features to another full connected neural network. Implementations in big data environments show promising performance of our model in handling big datasets with respect to feasibility and scalability.

In order to further improve the classification results and bridge the semantic gap between high-level concepts and low-level visual features, correlation discovery in semantic concept mining is worth exploring. Though inter-concept correlations have been utilized to address this issue recently, the very small number of instances in the minority classes often lead to the detection of imprecise correlations and unsatisfactory classification results. Meanwhile, correlation discovery is a computationally intensive task in the sense that it requires a deep analysis of very large and growing repositories. This dissertation further proposes a novel concept correlation analysis strategy framework that utilizes the correlations between the retrieval scores and labels. By integrating the correlation information, the proposed framework can help imbalanced data classification and enhance rare class (event or concept) mining even with trivial scores from the minority classes.

Not only deep learning but also numerous other classification algorithms have been developed for a variety of data types. However, it is nearly impossible for one classifier to perform the best in all kinds of datasets all the time. Therefore, ensemble learning models which aim to take advantages of different classifiers have received a

lot of attentions recently. In this dissertation, a scalable classifier ensemble framework assisted by a set of "judgers" is also proposed to integrate the outputs from multiple classifiers for multimedia big data classification. Specifically, based on the confusion matrices of different classifiers, a set of judgers are organized into a hierarchically structured decision model. A testing instance is first input to different classifiers, and then the classification results are passed to the proposed hierarchical structured decision model to derive the final result. The ensemble system can be run on Spark, which is designed for big data processing.

All the proposed components are evaluated on multimedia datasets containing different kinds of data. The experimental results show the effectiveness of our framework in classifying severely imbalanced data with promising performance, and demonstrate that the proposed classifier ensemble framework outperforms several state-of-the-art model fusion approaches. Furthermore, the proposed framework is applied to two real-world applications, i.e., deep learning based text data analysis on an Amazon review dataset and efficient large-scale stance Analysis in Twitter, and achieves promising results in both. In additional, we also design a web-based information retrieval system and identify several future directions that could be explored to further improve the current work.

To my dear parents and grandparents

Acknowledgements

First and foremost, I would like to express my sincerest appreciation to my dissertation supervisor and chairman of the committee Dr. Mei-Ling Shyu for her support and continuous guidance in my Ph.D. study. My deepest thanks also go to Dr. Shu-Ching Chen of the School of Computing and Information Sciences at Florida International University (FIU) for his encouragement and suggestions. Their professional and committed working attitude touches me and will continue to influence me. In addition, my great appreciation goes to Dr. Onur Tigli, Dr. Jie Xu and Dr. Hammam Alsafrjalani of the Department of Electrical and Computer Engineering at University of Miami for accepting the appointment to the dissertation committee and their consistent help.

Second, I am very grateful for the supporting through my Ph.D. study from the ECE department. It offers me teaching assistantships to multiple courses, which helps me develop my teaching techniques and communication skills. Moreover, my special thanks go to the Florida Department of Insurance for the financial support throughout my Ph.D. study. I would like to thank all my friends and colleagues in the Data mining, Database and Multimedia Research Group at UM and the Distributed Multimedia Information Systems Laboratory at FIU, for their long-term help and genuine support, especially when I met difficulty in research.

Last but not the least, I would like to extend my utmost gratitude to my parents and grandparents for their love and encouragement. It would have been impossible for me to finish this work without their consistent help and support. To them, I owe my eternal gratitude. My thanks also go to Dr. Kai Shen, Dr. Hong Qi, Dr. Shiyan Jiang, Mr. Qingzhou Zeng, Mr. Yang Liu, Mr. Wei Jiang, Ms. Ming Ma, Ms. Ni Yang, Ms. Yang Bai, Ms. Ye Yuan, Ms. Hongke Wang, Ms. Yuyang Chen and Ms. Xiuying Li. My beloved family and friends have always believed in me and have never

stopped offering their patient love and unconditional support. I would not have been able to conquer so many difficulties in my life without them.

Yilin Yan

University of Miami

August 2018

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Data imbalance is a challenging and common problem in machine learning, data mining, and information retrieval areas. A dataset is considered imbalanced when the data instances are not close to uniformly distributed across different classes. The classification of imbalanced datasets has recently attracted significant attentions due to its implications in several real-world use cases. The classifiers developed on datasets with skewed distributions tend to favor the majority classes and are biased against the minority classes. Despite extensive research interests, imbalanced data classification remains a challenge, especially for multimedia data. In the past decades, we have witnessed an explosion of multimedia data, thanks to the development of social media websites and blooming popularity of smart devices. As a result, multimedia semantic concept mining and retrieval whose objective is to mine useful information from the large amount of multimedia data including texts, images, and videos has become more and more important [1–25]. The huge amount of multimedia data and the semantic gap between low-level features and high-level semantic concepts have made it even more challenging. To address these challenges, it requires the joint research efforts from both big data mining and multimedia areas, and the correlations among the classes can provide important context cues to help bridge the semantic gap.

1.1 Motivations and Challenges

Skewness in data classes poses a significant challenge in major research problems pertaining to data mining and machine learning. Classes are rated as skewed or imbalanced when their data instances are non-uniformly associated to the class labels. In real-world cases, most applications have some degree of skewness inherently present in the data. Such datasets are often grouped into major and minor classes, where major classes have significantly greater numbers of instances associated with them as compared to minor classes. Some prominent imbalanced data use cases include fraud detection, network intrusion identification, uncommon disease diagnostics, and critical equipment failure. A number of famous classification methods are built to utilize the dataset statistics, which ends up being biased towards the majority classes. When identifying the minor classes, these classifiers often perform inaccurately even for very large datasets with considerable numbers of training instances.

Some notable frameworks aiming to solve this challenge were proposed before. The authors of these frameworks, along with others, targeted this issue from two different perspectives. The first type is algorithm-based approaches where the authors propose new frameworks or improve the existing methods using both supervised and unsupervised techniques. The second type is towards the manipulation of the data itself to reduce the skewness in the class attribution. However, the problem of imbalanced classes is far from being conquered, especially in multimedia data. Multimedia data is particularly difficult because of the various data types that are layered with spatio-temporal characteristics. Content-based multimedia data retrieval and management have become a very important research area due to its broad applications in this century [26–39]. For instance, video content analysis, in the context of auto-

matically analyzing human actions in videos, has been widely utilized in video event mining, video summarization, camera surveillance, etc. Meanwhile, the deluge of multimedia big data has made data-oriented research more and more important, especially in this big data era. Many data analysis technologies have been developed in the past decade, including a variety of classification algorithms for different kinds of datasets. Nevertheless, a single classifier can hardly handle heterogeneous multimedia data types from different datasets in various situations.

1.1.1 Deep Learning for Imbalanced Big Data Classification

Recently, one of the most popular paths to handle the challenging imbalanced data classification and retrieval for multimedia big data is to employ solutions from other domains of machine learning such as deep learning. Deep learning is the name of a whole family of algorithms that use graphs with multiple layers of linear and non-linear transformations to develop hierarchical learning models [40]. However, deep learning methods have not been used to address the data imbalance problem. As presented in [41, 42] on the TREC Video Retrieval (TRECVID) 2015 datasets, even the famous deep learning methods such as Convolutional Neural Network (CNN) which outperforms a multitude of conventional machine learning techniques face difficulties when dealing with the data imbalance problem.

Moreover, for big datasets in multimedia data mining, deep learning methods are very expensive on computations. For example, the method proposed in [43] took more than 30 days to train with 1755 videos, and the authors were only able to successfully train the deep learning framework using a near-duplicate algorithm. Although researches have paid extensive efforts on the data imbalance problem, rare

concept retrieval remains one of the most challenging problems in multimedia data [44–49].

1.1.2 Enhancing Rare Class Mining by Concept Correlation

While deep learning systems have achieved unprecedented progresses in a number of fields, concept correlations are mined and corresponding coefficients are utilized. Many approaches treat each concept as an individual class and convert one multi-concept detection problem into multiple binary classification problems [50]. Therefore, it ignores the correlations among different concepts. Nevertheless, the concepts are correlated in real-world multimedia datasets. For instance, some concepts co-occur more frequently, such as sky and cloud; while others rarely co-occur like road and fish. Such characteristics of correlations provide important context cues that can assist concept detection. Therefore, the calculation of correlations between concepts can help a lot in semantic concept mining and retrieval. There are many types of correlations, including Pearson correlation, Spearman correlation, and Cross correlation, which detect the number of times two things occur together [51].

One major challenge in correlation discovery is the huge volume of related datasets. With the rapid development of multimedia, communication, and Web 2.0 technology, massive amounts of multimedia data have been increasingly available on desktops and smart mobile devices via the Internet. Statistics shows that 72 hours of videos with all sorts of tags are uploaded to YouTube every minute and about 1.54 million photos are uploaded to Flickr every day [52]. Accordingly, the annual TRECVID competition organized by National Institute of Standards and Technology (NIST) has the "Semantic Indexing" task for concept detection from a large amount of videos

collected from Internet [53–57]. This calls for the need to implement the proposed framework in a big data environment.

1.1.3 Classifier Ensemble in Big Data Environment

To further improve the classification results on imbalanced datasets, a single classifier can hardly handle heterogeneous media types from different datasets in various situations. [58] proposed a fusion based classification system using statistical fusion such as Gaussian Mixture Model (GMM) fusion and Artificial Neural Network (ANN) fusion. Conflict results can be generated by different classifiers and the general idea to solve this is to find a way to fuse the results from different classification models. Previous results have indicated that the fusion of multiple different results can improve the performance of individual classifiers.

Another research topic in multimedia data is how to utilize multiple features from different feature extraction methods. In [59], the authors found the complementary nature of the descriptors from different viewpoints, such as semantics, temporal and spatial resolution. They also employed a hierarchical fusion that combines static descriptors and dynamic descriptors. In [60], textual features were shown to provide high-level semantics that are sometime difficult to be captured by visual features and a sparse linear fusion scheme was proposed to combine visual and textual features for semantic concept retrieval and data classification. Different classification frameworks can be employed for different kinds of features, which may discover different properties of the data [61, 62].

1.2 The Proposed Framework

Figure 1.1 shows the architecture of the proposed framework. It consists of three components which will be briefly discussed in the following subsections.

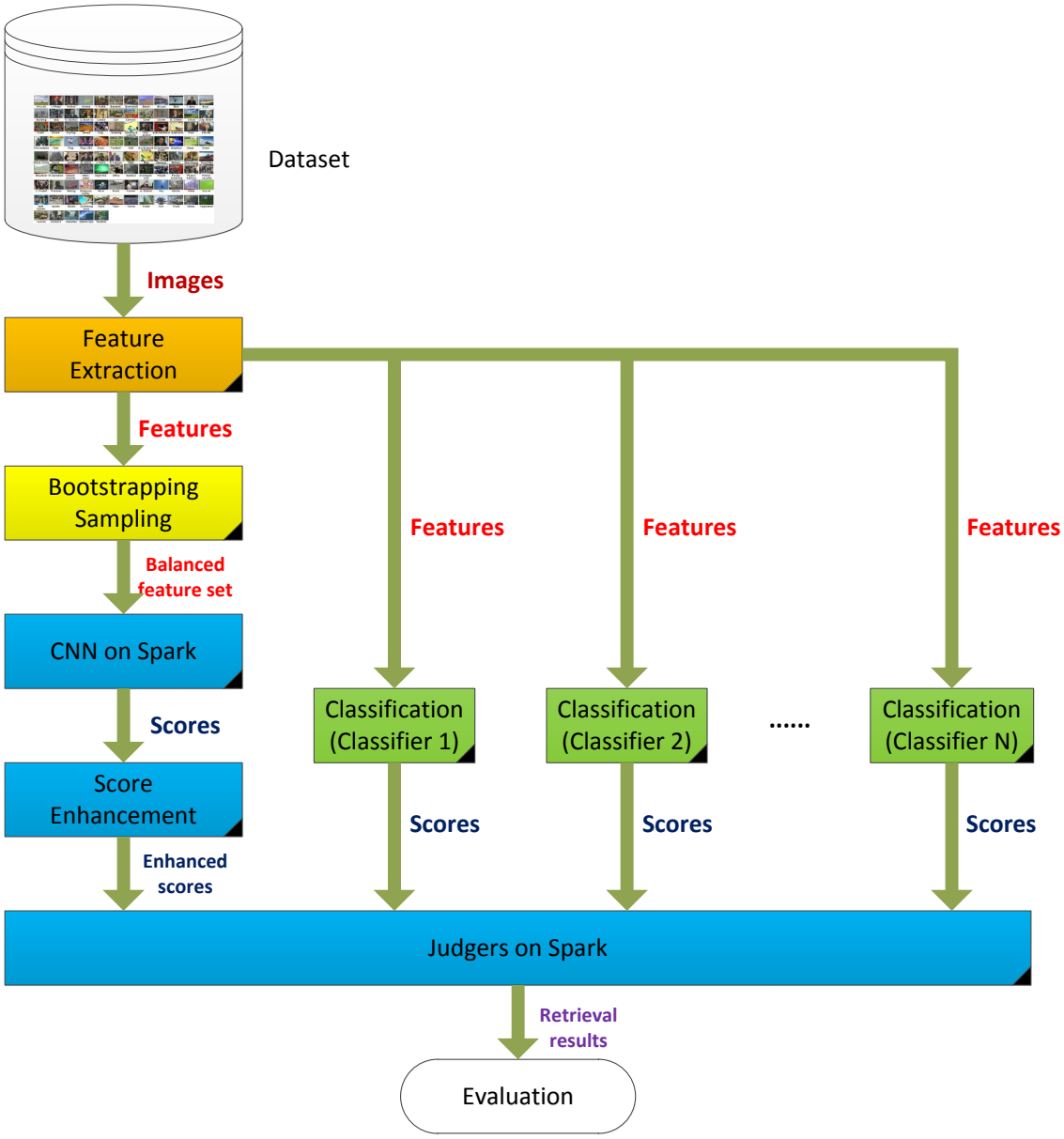


Figure 1.1: The proposed framework

1.2.1 The Imbalanced Concept Retrieval Component

In this dissertation, the framework proposed is for imbalanced concept retrieval in the field of multimedia big data management. Toward such demands, a method is proposed to help the classification of skewed data by a CNN based deep learning framework. Then, a big data deep learning approach coupled with a bootstrapping sampling technique is proposed to create a balanced set of batches using the training dataset. To the best of our knowledge, bootstrapping has not been used with the deep learning frameworks. To further facilitate the capability of handling the data imbalance problem in big datasets, a distributed computation framework using Apache Spark is also implemented to bind the novel qualities of CNN with the bootstrapping procedures. The proposed framework has shown to be highly impressive and comparatively economical in classifying highly skewed multimedia datasets. The Spark-based distributed computing capability enables a scalable architecture that can mine unstructured key-value confidence scores of multimedia data.

1.2.2 The Score Enhancement Component

Many concepts are often correlated, either positively or negatively. Some concepts co-occur rarely like cow and sea; while others co-occur more frequently such as bird and sky. Such correlations can provide important context cues to help detect the concepts [63–68]. While inter-concept correlations have been recently used to tackle the issue, the very small number of training instances in the minority class makes the task of correlation detection hard and often leads to unsatisfied concept retrieval results. Different from those enhancement models that only consider the correlations among concepts, we present a very different correlation analysis strategy that considers the

correlation between concept labels and retrieval scores. Even with trivial scores from minority classes, the proposed framework can enhance rare concept retrieval.

1.2.3 The Classifier Fusion Component

In the third component, a novel idea of classification combination is proposed in this dissertation. Generally speaking, an ensemble of data classifiers will be always better than the individual ones as "Vox Populi, Vox Dei". To take advantages of different classifiers and reach the best performance on a dataset, lots of research groups recently focus on assembling useful classifiers together. Based on the confusion matrices of different classifiers, a scalable classifier ensemble framework assisted by several "judgers" is proposed to integrate the outputs from multiple classifiers for multimedia big data classification. These judgers are put together as a boosted classifier. Specifically, a set of "judgers" are generated based on the training and validation results from different classifiers and features at first. On the second step, these judgers are ranked and organized into a hierarchically structured decision model. Finally, an Apache Spark-based classification system is developed which can be used for multimedia big data classification.

1.3 Evaluation Metrics

1.3.1 Confusion Matrix

In general, a classifier is evaluated by a confusion matrix as illustrated in Table 1.1. The columns are the predicted class and the rows are the state of nature (actual class). In the confusion matrix, TP (True Positives) and FP (False Positives)

represent the numbers of positive data instances that are correctly or incorrectly classified, respectively. Similarly, TN (True Negatives) and FN (False Negatives) indicate the numbers of negative data instances being correctly or incorrectly classified, respectively. For performance comparison, the precision and recall metrics [69] are commonly used and are derived from the confusion matrix in Table 1.1 and Equations (1.1) and (1.2).

Table 1.1: Confusion matrix

State of nature \ Prediction	Positive	Negative
Positive	True Positives (TP)	False Negatives (FN)
Negative	False Positives (FP)	True Negatives (TN)

1.3.2 Precision, Recall, and F-score

The recall and precision goals can often be conflicting, since the increase of true positive data instances for the minority class may also increase the number of false positive data instances, which will reduce the precision. For imbalanced data classification, the recall value is normally considered a more important criterion because it is more desirable to detect as many interesting events as possible, even at the expense of adding a reasonable number of false positive data instances. For example, users often want to locate all possible frauds in banking operations followed by a manual double check to root out false alarms, instead of missing true scams. In addition, F-score, also known as F1 measurement or F-value, captures the trade-offs between precision and recall, and is considered an objective and ultimate quality metric of a classifier which is defined in Equation (1.3).

$$precision = \frac{TP}{TP + FP} \tag{1.1}$$

$$recall = \frac{TP}{TP + FN} \tag{1.2}$$

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{1.3}$$

1.3.3 Mean Average Precision

The average precision (AP) value, a widely used metric in the multimedia concept retrieval domain, is also used in the evaluation. For a given concept, $Pre(i)$ indicates the precision of the i^{th} data instance in the ranking list. ψ is the number of the retrieved data instances; while G_n is the total number of data instances containing that concept in the database. $Min(G_n, \psi)$ indicates the smaller value of G_n and ψ. The average precision at ψ (i.e., $AP@\psi$) is defined in Equation (1.4). By generating the AP values for all the target concepts and calculating the mean value of them, the mean average precision (MAP) value is used to capture the ranking information.

$$AP@\psi = \sum_{i=1}^{\psi} \frac{Pre(i) \times rel(i)}{Min(G_n, \psi)}, \tag{1.4}$$

$$\text{where } rel(i) = \begin{cases} 1, & \text{if instance } i \text{ is positive;} \\ 0, & \text{otherwise.} \end{cases}$$

1.4 Contributions and Limitations

Several contributions are made in this dissertation on the topic of imbalanced data classification and information retrieval for multimedia big data:

1. Developed a CNN based deep learning solution integrated with a bootstrapping technique to overcome the data imbalance problem.

2. Proposed to extract features from pre-trained CNN models and feed those features to another full connected neural network considering the fact that CNNs are very computationally expensive coupled with big training datasets.

3. Designed an efficient multimedia rare concept retrieval model by constructing a semantic concept hierarchy and using concept correlations.

4. Novel concepts of positive and negative "judgers" are defined to assemble a novel hierarchical structured decision framework. Meanwhile, the proposed framework considers the fusion of classifiers using different kinds of features.

5. A Spark implementation of the proposed framework shows promising performance of the designed model in handling big datasets with respect to feasibility and scalability.

6. Two novel applications, including a very unique one for efficient large-scale twitter stance analysis, are designed and implemented.

Meanwhile, the dissertation has the following assumptions and limitations:

1. Since deep learning has been growing very fast, many new algorithms and new architectures appear every few months, which is out of the scope of this article.

2. The framework only focuses on multimedia data, while deep learning is now being used in a wide range of applications. More kinds of applications could be considered in future work such as autonomous driving, data compression, outlier detection, Biomedicine, disaster management, etc.

3. Some parameters in the framework are based on an iterative search on the training data to find the optimum values. This empirical approach may be

inevitable, but in some cases an advanced parameter estimation approach can be investigated.

1.5 Organization of the Dissertation

This dissertation is organized as follows. Chapter 2 reviews the techniques related to imbalanced data classification, correlation coefficients, hierarchical models, deep learning, big data technology, and fusion of multiple classifiers and features. The advantages and limitations of peer work are analysed and discussed. Chapter 3 introduces the deep learning based imbalanced multimedia concept retrieval component in the big data environment. Chapter 4 presents the component of correlation-assisted concept retrieval score enhancement. In Chapter 5, the classifier fusion and score integration framework by judgers on Spark clusters is introduced. By applying the three proposed components, Chapter 6 shows two applications in text data analysis and a web-based information retrieval system. As the extensions of the existing framework, conclusions are drawn in Chapter 7, which also propose several future research directions.

CHAPTER 2

Literature Review

In this chapter, the literature review on the related work based on the framework of imbalanced data classification and information retrieval for multimedia big data is presented. The thorough review covers techniques in the area of deep learning, imbalanced data classification, big data technology, correlation coefficients, as well as fusion of multiple classifiers and features. Recent research directions as well as popular algorithms in these areas are introduced in this chapter. Furthermore, the advantages and limitations of peer work are analysed and discussed.

2.1 Deep Learning

In recent years, machine learning has become more and more popular in research and a large number of applications, including multimedia concept retrieval, image classification, video recommendation, social network analysis, text mining, etc. Among various machine learning algorithms, "deep learning", also known as representation learning [70], is widely used in these applications nowadays. With great successes around many fields, deep learning now is one of the hottest research directions in the machine learning society. This section gives an overview of deep learning from different perspectives, and the novelty is that it focuses on different aspects of

deep learning with a review of top-level papers, the authors' experience and discovery in research on neural networks, as well as the experience from the researchers.

2.1.1 Why Deep Learning

Similar to how medicine, energy, transportation, manufacturing, industrialization, and food production were revolutionized by electricity in the 20th century, deep learning is poised to set course in becoming part of the next century of revolutions. Moreover, the explosive growth and availability of data as well as the remarkable advancement in hardware technologies have led to the emergence of new studies in deep learning. Deep learning which has its root from conventional neural networks significantly outperforms its predecessors. It utilizes graph technologies with transformations among neurons to develop many layered learning models. Carnegie Mellon University (CMU) even has a single machine learning department. Many latest deep learning techniques have been presented and demonstrated promising results across different kinds of applications such as Natural Language Processing (NLP), visual data processing, speech and audio processing, and many other well-known applications [49, 68, 71].

Traditionally, the efficiency of machine learning algorithms highly relied on the goodness of the representation of the input data. A bad data representation often leads to lower performance compared to a good data representation. Therefore, feature engineering has become an important research direction in machine learning for a long time, which focuses on building features from raw data and has led to lots of research studies. Furthermore, feature engineering is often very domain specific and requires significant human efforts. For example, in computer vision, different kinds

of features have been proposed and compared, including Histogram of Oriented Gradients (HOG) [72], Scale Invariant Feature Transform (SIFT) [73], and Bag of Words (BoW). Once a new feature is proposed and performs well, it would become a trend for years. Similar situations happened in other domains including speech recognition and NLP.

Comparatively, deep learning algorithms perform feature extraction in a quite automated way which allows the researchers to extract discriminative features without domain knowledge and human input [74]. These algorithms include a layered architecture of data representation, where high-level features can be defined in the top layers while low-level features are extracted from the bottom layers. These kinds of architectures are originally inspired by artificial intelligence (AI) simulating its process of the key sensorial areas in the human brain. The brains never learn feature extraction algorithms like "SIFT," but can automatically extract features from different scenes. The input is the scene information received from eyes; while the output is the classified objects. This highlights the major advantage of deep learning: works like real human brains.

2.1.2 History

While deep learning is being widely used in different domains from speech and image recognition, to NLP and industry-focused implementations such as fraud detection and recommendation systems, the history of neural networks is unbelievably long. Building a machine which can simulate human brains had been a dream of sages for centuries. The very beginning of deep learning can be traced back to 300 B.C. when Aristotle proposed "associationism," which started the history of human's

ambition trying to understand the brain, since such an idea requires the scientists to understand the mechanism of human recognition systems. The modern history of deep learning started in 1943 when the McCulloch-Pitts (MCP) model was introduced and known as the prototype of artificial neural models [75]. They created a computer model based on the neural networks functionally mimicking neocortex in human brains [76]. The combination of the algorithms and mathematics called "threshold logic" was used in their model to mimic the human thought process but not to learn. Since then, deep learning has evolved steadily with a few significant milestones in its development.

After the MCP Model, the Hebbian theory, originally for the biological systems in the natural environment, was implemented [77]. After that, the first electronic device called "perceptron" with the context of the cognition system was introduced in 1958, though it is different from typical perceptrons nowadays. The perceptron highly resembles the modern ones which have the power to substantiate "associationism." At the end of the first AI winter, the emergence of "backpropagandists" became another milestone. Werbos introduced backpropagation, the use of errors in training deep learning models, which opened the gate to modern neural networks. In 1980, "neocogitron" which inspired the CNN was introduced [78]; while Recurrent Neural Networks (RNNs) were proposed in 1986 [79]. Next, LeNet made the deep neural networks work practically in the 90s, however it did not get highly recognized [80]. Due to the hardware limitation, the structure of LeNet is quite easy and cannot be applied to large datasets.

Around 2006, Deep Belief Networks (DBNs) along with a layer-wise pre-training framework were developed [81]. Its main idea was to train a simple two-layer unsuper-

vised model like Restricted Boltzman Machines (RBMs), freeze all the parameters, stick on a new layer on top, and train just the parameters for the new layer. Researchers were able to train neural networks that were much deeper than the previous attempts using such a technique, which prompted a rebranding of "neural networks" to "deep learning". Originally from Artificial Neural Networks (ANNs) and after decades of development, deep learning now is one of the most efficient tools comparing to other machine learning algorithms with great performance. We have seen a few deep learning methods rooted from the initial ANNs, including DBNs, RBMs, RNNs, as well as the CNNs [43, 82].

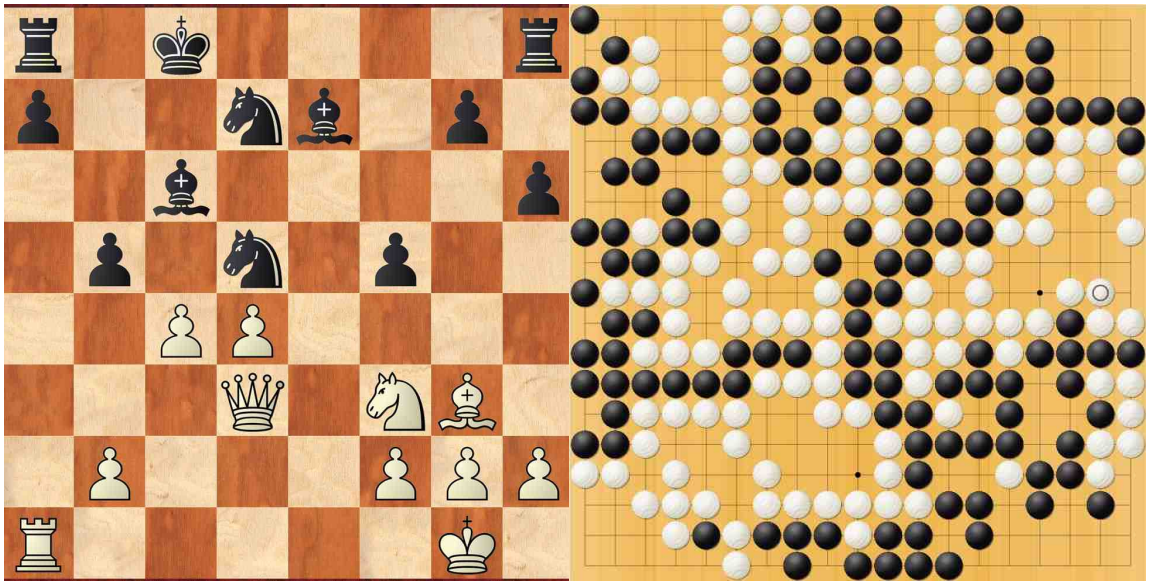
2.1.3 The Astonishment from AlphaGo

With no doubt, Google AlphaGo completely shocked the world at the start of year 2017 [83]. Under the pseudonym name "master," it won 60 straight online games against human professional Go players from Dec. 29th, 2016 to Jan. 4th, 2017. In just a week, AlphaGo's online record was 60 wins and 0 losses, including three victories over Ke Jie, an 18-year-old recognized as the world's best Go player. Though deep learning had been the most popular research topic in machine learning for some years, it was limited in the computer science society. The great success of AlphaGo completely shows the power of deep learning to public as Go is quite a popular game and the 2016 matches were watched by perhaps a hundred million people worldwide. After the New Year break, thousands of media were reporting the news around the world from the Wall Street Journal to China Daily, and millions of related comments were posted on social media with hashtag "AlphaGo." Ke initially claimed that he would be able to beat AlphaGo, but declined to play against it at

first for fear that it would "copy his style." As the matches progressed, Ke went back and forth, even lost his confidence after analyzing the first three lose matches, though regaining confidence after AlphaGo displayed flaws in the fourth match. However, the AlphaGo AI could be modified after each match and the "bad steps" are unlikely happen again in the next match.

This is not the first time when humans are beaten by a computer in games. Back to the prior generation in May 11th 1997, IBM's chess-playing computer "Deep Blue" won the six-game rematch 3.5-2.5 versus Garry Kasparov, whom many consider to be the greatest chess player of all time. Figure 2.1 shows the final position after "*19.c4*" in Game 6. While Deep Blue lost a six-game match on Feb 10th 1996, it finally defeated Kasparov just in the next year. The results astonished the world since the public didn't believe computers can "think" like real human beings though IBM refused a rematch again and retired Deep Blue. Twenty years later, while IBM is no longer the most noticeable computer science corporation, the News of AlphaGo from Google shocked the world again. Since the Go board is on a 19 by 19 array, the player who starts a match has 361 possible empty points to place his or her stone mathematically, leaving 360 empty points for his or her opponent's next move [84]. This goes on to searching the factorial of 361 situations for the best results. Thus, the Go board has many more options than a chess board which has only 8 by 8 possible combinations. Moreover in a chess match, a chess piece's move is confined to its next step, as the rook can only move horizontally or vertically, which significantly reducing its number of combinations. Therefore, most Go players don't believe super machines can defeat them after the win of Deep Blue 20 years ago. Although the evolution

of computing makes CPU's frequency 160 times faster than 20 years ago, the huge number of options in Go is still beyond computing power.



(a) Deep Blue versus Kasparov, Game 6 (b) AlphaGo versus Lee Sedol, Game 5

Figure 2.1: Deep Blue in a Chess game and AlphaGo in a Go game

While human Go players need to estimate a territorial advantage on the board, the AlphaGo does not even need to actually understand the "rule" of Go and can evaluate a position by applying deep learning algorithms. In other words, deep learning algorithms can break the barrier of "human instinct" which other algorithms cannot capture. All in all, AlphaGo is able to defeat world champion Go players because it uses the modernest deep learning algorithms and sufficient hardware resources. The AlphaGo mainly contains three layers including a supervised learning network, a value network, and a reinforcement learning network [85]. Among them, the supervised learning network and the value network were used during single matches; while the reinforcement learning network was responsible for reinforcement training. In a six-game match, the reinforcement training could even be applied to the first game

and would improve AlphaGo in the following games. The supervised learning network learns from a board position in games and mimics where the champion players would play for the next step based on the stored 30 million matches in its database. Finally, AlphaGo used a mixture of the output from its value network to place stones on the Go boards.

2.2 Imbalanced Data Classification

In general, imbalanced data classification techniques fall into three categories, namely sampling-based, algorithm-based, and feature selection-based approaches.

2.2.1 Sampling Based

The most popular classification algorithms for imbalanced datasets are sampling-based approaches. Oversampling and undersampling methodologies have received significant attentions to counter the effect of imbalanced datasets. Studies have tested different variants of oversampling and under-sampling techniques, and presented (sometimes conflicting) viewpoints on the usefulness of oversampling versus down-sampling [86] for imbalanced datasets.

In general, downsampling is to select a part of negative samples (data instances) to build a model with a similar number of positive samples. It is very efficient as it uses only a subset of the majority class. The main disadvantage is that many data instances in the majority class are ignored and may result in loss of information. Liu et al. proposed two algorithms to overcome this deficiency [87]. “Easy Ensemble” samples several subsets from the majority class, trains a classification model using each of them, and integrates the out-puts of those models to produce the final predication

results. "Balance Cascade" trains the models sequentially. In each step, the majority class data instances that are correctly classified by the current trained models are removed from the next round.

In terms of oversampling, duplicate or similar positive data instances are generated by certain algorithms to make the dataset balanced. Zhang et al. presented an improved over-sampling approach based on the synthetic minority over-sampling technique (SMOTE) [88]. First, data distribution of the minority class is used to estimate whether different types of data instances are overlapped. Next, synthetic data instances are generated in different classes when classes overlap significantly with each other. In addition, weights are increased for those positive samples that are far from the borderline. However, oversampling can potentially lead to overfitting.

2.2.2 Algorithm Based

The common goal of algorithm-based approaches is to optimize the performance of learning algorithms on unseen data to address the class/data imbalance problem. One-class learning methods recognize the data instances belonging to a specific class and reject the others. Under certain conditions, such as in a multi-dimensional dataset, one-class learning achieves better performance than the peers [89]. Cost-sensitive learning methods try to maximize the loss functions associated with a dataset to improve the classification performance. These learning methods are motivated by the observation that most real-world applications do not have uniform costs for misclassifications. The actual costs associated with each kind of errors are unknown typically, so these methods need to determine the cost matrix based on the data

and apply it to the learning stage. A closely related idea to cost-sensitive learners is shifting the bias of a machine to favor the minority class.

Genetic Algorithm based Selective Ensemble Network (GASEN) has been proven very effective to select a subset of neural networks to form an ensemble classifier or a regressor of the enhanced generation ability. Che et al. tested GASEN on dozens of datasets and found that there is some potential for improving GASEN's performance on data imbalance learning [90]. However, such studies on GASEN are far from extensive or systematic. Machine learning algorithms, such as Genetic Programming (GP), can also generate biased classifiers when the datasets are imbalanced. Bhowan et al. used new fitness functions in the GP learning process and empirically showed better performance by the evolved classifiers on both minority and majority classes [91].

2.2.3 Feature Selection Based

The goal of feature selection, in general, is to select n features from a feature set that allow a classifier to reach an optimal performance, where n is a user-defined parameter. As a key step for many machine learning and data mining algorithms especially for high-dimensional datasets, feature selection has been thoroughly studied, where filters are used to score each feature independently based on a rule [92]. However, its importance in resolving the data imbalance problem is a recent direction [93]. This direction is motivated by the fact that in real-life data, the data imbalance problem is commonly accompanied with the issue of high data dimensionality which both sampling techniques and algorithm-based approaches may be insufficient to deal with [89]. Therefore, a number of research work has been conducted to per-

form feature selection to tackle the data imbalance problem recently. For example, Ertekin [94] studied the performance of feature selection metrics in classifying text data drawn from the Yahoo Web hierarchy. They applied nine different metrics and measured the power of the best features using the Naive Bayes (NB) classifier.

Wasikowski et al. presented the first systematic comparison of different approaches using seven feature selection metrics. They evaluated the performance of these metrics based on the Receiver Operating Characteristic (ROC) curve and Area Under the precision-recall Curve (AUC) [89]. Jamali et al. discussed a prior knowledge for an expert system, which can identify the best performed feature selection metric based on the data characteristics regardless of the classifier used [95]. Zheng et al. investigated the usefulness of explicit control of combination within a proposed feature selection framework using multinomial Naive Bayes and regularized logistic regression as classifiers [96].

2.3 Big Data Technology

A big data processing platform not only needs a cluster computing infrastructure, but also requires complementary tools for efficient calculations since an outdated component could be the bottleneck of the platform. In the past 5 years, many corresponding tools have been developed to support the big data era. In this dissertation, an efficient processing platform for negative correlation discovery is built with the utilization of some of these tools.

2.3.1 NoSQL Database

A Database Management System (DBMS) is a software system that enables users to define, create, maintain, and control access to the database based on a database model. In the past half-century, a Relational Database Management System (RDBMS) using Structured Query Language (SQL) has been a dominant solution. Relations bring the benefits of group-keeping the data as constrained collections (in tables) containing the information in a structured way, and relate all the inputs by assigning values to the attributes. During the past decades, database systems that implemented the relational models are more and more efficient and reliable, e.g., MySQL, PostgreSQL, and SQLite. However, with the rapid development of big data computing techniques, the traditional relational data model faces great challenges when working with other big data processing tools and often comes out as the bottleneck of the infrastructure. In particular, when the size of a relational table increases tremendously, even answering simple queries becomes a problem.

The recent development has made many implementations available though each works very differently and serves a specific need. These schema-less solutions either allow an unlimited forming of entries or have a very simple but extremely efficient key-based value stores. For example, the NoSQL database systems do not come with a model as used with the structured relational solutions. Different from most people think, NoSQL databases actually have existed since the 1960s, but have recently gained attractions with popular options such as MongoDB, CouchDB, Redis, and Apache Cassandra [97]. Among them, Cassandra is well-known for its high-availability and high-throughput characteristics and is capable of handling enormous write loads and surviving cluster node failures. In terms of the Consistency, Availabil-

ity, and Partition tolerance (CAP) theorem, Cassandra provides tunable consistency and availability for operations. What is more interesting when it comes to data processing is that Cassandra is linearly scalable and it provides cross-data center replication capabilities. In the proposed processing platform, the key-value pair can be directly stored in Cassandra since it supports multi-values in one column.

2.3.2 Apache Hadoop

Hadoop MapReduce [98] is a software framework for easily writing applications that process vast amounts of data in parallel on thousands of nodes of commodity hardware in a reliable and fault-tolerant manner. Hadoop [99], its open-source implementation, allows the distributed processing of large datasets across the clusters of computers using simple programming models. It is designed to scale up from a single server to thousands of machines, each offering local computation and storage. Rather than relying on hardware to deliver high-availability, Hadoop is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures. Hadoop provides a distributed file system called Hadoop Distributed File System (HDFS). It splits the input files into large blocks and distributes them amongst the nodes in the cluster. To process the data in parallel, Hadoop MapReduce transfers the packaged code to nodes based on the data each node needs to process. A MapReduce program consists of two user-defined functions: a map function to process pieces of the input data (called input splits), and a reduce function to aggregate the output of invocations of the map function. Both functions use user-defined key-value pairs as the input and output.

2.3.3 Apache Spark

Spark is an open source big data processing framework advertised as "lightning" fast cluster computing built around speed, ease of use, and sophisticated analytics [100]. It provides a faster and more general data processing platform. The Spark core is complemented by a set of powerful, higher-level libraries which can be seamlessly used in the same application. These libraries currently include SparkSQL, Spark Streaming, MLlib (for machine learning), and GraphX. Additional Spark libraries and extensions are currently under development as well. Spark introduces the concept of an Resilient Distributed Dataset (RDD), an immutable fault-tolerant, distributed collection of objects that can be operated in parallel. An RDD can contain any type of objects and is created by loading an external dataset or distributing a collection from the driver program.

Spark has several advantages compared to other big data and MapReduce technologies including Hadoop. For example, Spark's multi-stage in-memory primitives provide performance up to 100 times faster for certain applications [100–102] when comparing to Hadoop's two-stage disk-based MapReduce paradigm. In addition to the Map and Reduce operations, Spark provides many operations called transformations such as map, flatMap, sample, filter, groupByKey, reduceByKey, union, join, sort, cogroup, mapValues, and partionBy. Developers can use these operations standalone or combine them to run in a single data pipeline use case. Moreover, Spark can run programs up to 100x faster in memory or 10x faster on disk than Hadoop.

Both Hadoop and Spark are built on Yet Another Resource Negotiator (YARN). In Hadoop 2.0, YARN is split from MapReduce and runs on top of it. YARN is a generic cluster resource management framework that can run applications on a

Hadoop cluster. In the YARN model of computation, ResourceManager runs as a master daemon and manages ApplicationMasters and NodeManagers. ApplicationMaster is a lightweight process that coordinates the execution of tasks of an application and requests the resource containers for tasks from the ResourceManager with the NodeManager offering the resources (memory and CPU) as the resource containers.

Mesos is a distributed system kernel which essentially uses a container architecture but is abstract enough to allow a seamless execution of multiple (sometimes identical) distributed systems on the same architecture, minus the resource overhead of the virtualization systems [103]. This includes an appropriate resource isolation while still allowing data locality needed for frameworks like MapReduce. Mesos was built to be a global resource manager for the entire data center. The primary difference between Mesos and YARN is their schedulers. In Mesos, when a job comes in, a job request comes into the Mesos master, and what Mesos does is to determine what the resources are available and to make the offers back. Those offers can be accepted or rejected. This allows the framework to decide what the best fit is for the job that needs to be run. If Mesos accepts the job for the resources, then it places the job on the slave and all is done. It has the option to reject the offer and wait for another offer to come in. One big advantage of Mesos over YARN is that it can manage all the resources in the data center. Therefore, Spark is run on Mesos instead of YARN in our proposed system. Mesos cluster consists of Master nodes which are responsible for resource offers and scheduling and Slave nodes which do the actual heavy lifting of task executions.

2.4 Correlation Coefficients and Re-ranking Strategies

A correlation coefficient is a coefficient that illustrates a quantitative measure of the statistical relationships between two or more random variables or observed data values. Correlations can be divided into the following four different types (namely nominal, ordinal, interval, and ratio) based on the category of the input data. The overall of the operations and scale measures is shown in Table 2.1.

Table 2.1: Overall of data types and scale measures

Scale Measures / Operations	Nominal scales	Ordinal scales	Interval scales	Ratio scales
Frequent distribution	Yes	Yes	Yes	Yes
Mode	Yes	Yes	Yes	Yes
Median	Yes	Yes	Yes	Yes
Mean	No	Yes	Yes	Yes
Plus	No	No	Yes	Yes
Minus	No	No	Yes	Yes
Multiple	No	No	No	Yes
Divide	No	No	No	Yes

2.4.1 Nominal Scale

Nominal scales are used to label variables that do not have quantitative values, and the data are put into categories without any order or structure. For example, any data with the YES/NO labels is nominal since it has no order and there is no distance between YES and NO. Another example is the data of colors that the underlying spectrum is ordered but the names are nominal.

There are many coefficients in this category. Given a 2×2 contingency table, and let O_i be an observed frequency, E_i be an expected (theoretical) frequency asserted by the null hypothesis, and n be the number of cells in the table. χ^2 [104] is the Pearson's

cumulative test statistics, which asymptotically approaches a χ^2 distribution as shown in Equation (2.1). If N is the grand total of the observations, the coefficient ϕ can be defined in Equation (2.2). This coefficient can only be calculated for frequency data represented in 2×2 tables.

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}. \tag{2.1}$$

$$\phi = \sqrt{\frac{\chi^2}{N}}. \tag{2.2}$$

The Contingency (C) coefficient and Cramer's V coefficient are similar. The C Coefficient [105] in Equation (2.3) is used when there are 3 or more values for each nominal variable, as long as there are an equal number of possible values leading to the construction of a data matrix that has the same numbers of rows and columns (e.g., 3×3, 4×4, etc.). C suffers from the disadvantage that it does not reach a maximum of 1 or the minimum of -1. The highest it can reach in a 2×2 table is 0.707, and the maximum it can reach in a 4×4 table is 0.870. It can reach values closer to 1 in the contingency tables with more categories. It should, therefore, not be used to compare associations among tables with different numbers of categories. Moreover, it does not apply to asymmetrical tables (i.e., those with different numbers of rows and columns). On the other hand, Cramer's V coefficient in Equation (2.4) is used when the numbers of possible values for the two variables are not the same, yielding different numbers of rows and columns in the data matrix (e.g., 2×3, 3×5, etc.). It is also a measure of associations between two nominal variables, giving a value between 0 and +1 (inclusive). Cramer's V coefficient [106] is widely used because it can solve both multiply variable cases and asymmetrical variable cases, but it can be a heavily biased estimator of its population counterpart and may tend to overestimate

the strength of an association.

$$C = \sqrt{\frac{\chi^2}{\chi^2 + N}}; \tag{2.3}$$

$$V = \sqrt{\frac{\chi^2}{N(k-1)}} = \sqrt{\frac{\phi^2}{(k-1)}} = \sqrt{\frac{\phi^2}{\min[(r-1),(c-1)]}}. \tag{2.4}$$

2.4.2 Ordinal Scale

The order of the values in ordinal scales is more significant, so smaller (<) and bigger (>) can be applied to the values but the differences between them is not really known. However, an ordinal scale can only interpret a gross order but not the relative positional distances. The simplest example is ranking, and there is no objective distance between any two points on the subjective scale. The top may be far superior to the second in one case; while the distance may be subjectively small in another case.

There are several coefficients in this category. The Gamma (G) coefficient [107] is for symmetrical correlation from -1 to +1 as follows:

$$G = \frac{N_s - N_d}{N_s + N_d}, \tag{2.5}$$

where N_s is the parity of the number of non-inversions, i.e., the pairs of elements x, y of σ such that $x<y$ and $\sigma(x)<\sigma(y)$ or $x>y$ and $\sigma(x)>\sigma(y)$; N_d is the parity of the number of inversions, i.e., the pairs of elements x, y of σ such that $x<y$ and $\sigma(x)>\sigma(y)$ or $x>y$ and $\sigma(x)<\sigma(y)$. In this case, N_s is the number of pair of cases that are ranked the same on both variables; while N_d is the number of pair of cases that are ranked differently on both variables. The Gamma coefficient needs all nominal variables to be ranked. A similar one is the Kendall tau (τ) coefficient which considers the tied pairs.

The Somers d coefficient [104] is for asymmetrical correlation. If X is an independent variable (column) and N_y is the parity of the number of non-inversions in rows, versus if Y is an independent variable (row) and N_x is the parity of the number of non-inversions in columns, then the Somers d coefficients can be defined as follows:

$$d_{xy} = \frac{N_s - N_d}{N_s + N_d + N_y}; \tag{2.6}$$

$$d_{yx} = \frac{N_s - N_d}{N_s + N_d + N_x}. \tag{2.7}$$

The Spearman's rank correlation coefficient [108] is a nonparametric measure of statistical dependence between two variables. It assesses how well the relationship between two variables can be described using a monotonic function. If there are no repeated data values, a perfect Spearman correlation of -1 or +1 occurs when each of the variables is a perfect monotone function of the other. The Spearman correlation coefficient is defined as the Pearson correlation coefficient between the ranked variables. For a sample of size n, the n raw scores X_i and Y_i are converted to ranks x_i and y_i, and ρ is computed from Equation (2.8). Identical values (such as rank ties or value duplicates) are assigned a rank equal to the average of their positions in the ascending order of the values.

$$\rho = \frac{\sum_{i}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i}^{n} (x_i - \overline{x})^2 (y_i - \overline{y})^2}} \tag{2.8}$$

2.4.3 Interval Scale

Data in this category are numeric scales in which not only the order but also the exact differences between the values are known and thus the realm of statistical analysis on such data opens up. The Celsius temperature is considered the classic

example data in this category. Furthermore, since they are numeric variables, plus (+) and minus (−) can also be applied.

The Pearson product-moment correlation (ρ or r) coefficient [109, 110] is an example coefficient in this category. It is a measure of the linear correlation dependence between two variables X and Y, giving a value between -1 and +1 (inclusive), where -1 is a total negative correlation, 0 is no correlation, and +1 is a total positive correlation. It is widely used in sciences as a measure of the degree of the linear dependence between two variables. For a population:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}, \tag{2.9}$$

where *cov* is the covariance of X and Y and is equal to $E\{[x-E(x)][(y-E(y)]\} = E(xy) - E(x)E(y)$; σ_X is the standard deviation of $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu_X)^2}$; σ_Y is the standard deviation of $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-\mu_Y)^2}$; μ_X is the mean of X (i.e., $\frac{1}{N}\sum_{i=1}^{N}x_i$); μ_Y is the mean of Y (i.e., $\frac{1}{N}\sum_{i=1}^{N}y_i$); and finally E is the expectation.

2.4.4 Ratio Scale

Data in the ratio scales have the order of the values, the exact value between units, and an absolute zero. Thus a wide range of both descriptive and inferential statistics can be applied to the data in the ratio scale. Since they are numeric variables with an absolute zero (like temperature, mass, etc.), the multiply (×) and divide (÷) operators can also be applied. However, data in multimedia and social research are usually not in this category, but most of the correlation coefficients for the interval scales can also be applied to data in the ratio scales.

2.4.5 Integrated Correlation Factor

While a number of approaches have adopted positive correlations to improve the performance of semantic concept mining and retrieval, very few work is explored for negative correlations among concepts. Based on the literature review, finding negative correlations in the multimedia big data area is a challenge. Some studies speculated that negative correlations might only improve the performance slightly [111]; while some other groups even reported their performance gain by negative correlations was nearly 0.

Unlike positive instances, the labels of a large number of negative instances are actually inferred in many large-scale multimedia datasets. This is partially because when the label of a training sample is manually annotated as "skipped" or "not sure", we usually consider it as a negative instance. Although the co-occurrence of two concepts in a video shot (keyframe) increases the probability that they are positively correlated, one concept does not occur while the other appears does not necessarily mean that they are negatively correlated. It is hard to conclude the appearance of the concept "bird" suppresses the appearance of the concept "flower" just because they do not co-occur together.

Generally speaking, if two concepts are negatively correlated, their correlations would not be affected by the existence of the third concept (called a control concept in this dissertation). Integrated Correlation Factor (ICF) represents an average quantitative metric of correlations under different control concepts. Let Ω be the set of all concepts, $|\Omega|$ be the total number of concepts, and C_D represent the control concept. Similarly, with the definitions of C_T^+ and C_R^+, C_D^+ is the condition that a data instance is positive for C_D. Using such information, we define ICF between the target concept

and the reference concept in Equation (2.10).

$$ICF(T,R) = \frac{1}{|\Omega| - 2} \sum_{D \in \Omega, D \neq T, D \neq R} \rho(C_T, C_R | C_D^+), \tag{2.10}$$

where $\rho(C_T, C_R|C_D^+)$ is the Pearson product-moment correlation coefficient [112] between C_T and C_R given C_D^+, which has been explained in Section 2.4.3.

2.4.6 Hierarchical Models

Many research efforts have been conducted on organizing the hierarchies for semantic concept retrieval and event detection. Most of them use inter-concept correlations to build the hierarchies. Wang et al. [113] proposed a hierarchical context model to systematically integrate feature level context, semantic level context, and prior level context for accurate and robust event recognition in surveillance videos. A comprehensive model that can integrate contexts from all three levels simultaneously was built. In [114], the authors presented a large-scale video event classification system with a large number of event categories mined automatically from YouTube video titles and descriptions using Part-of-Speech parsing tools, with constraints derived from WordNet hierarchies. To solve the problem of multi-class object detection, the authors proposed a boosted multi-class object cascade that only splits one class object from the upper-level cascade when building the sub-cascades [115], which reduces the number of classifiers in each stage. Vreeswijk et al. [116] analysed the differences between the images labelled at varying levels of abstraction and the union of their constituting leaf nodes.

Recently, some researchers find that inter-concept correlations can help re-rank the concept detection scores on event detection. The selection of event-specific concepts based on the similarity to a textual event description had shown to yield effective

event detection results without positive examples [117]. Tao et al. [118] showed that inter-concept associations including both positive and negative correlations can be used to bridge the semantic gap and enhance the performance of semantic concept detection in multimedia data [119–122]. The concept-concept association information integration and multi-model collaboration framework were proposed to enhance high-level information retrieval from multimedia big data.

2.5 Fusion of Multiple Classifiers and Features

In the scope of multimedia data classification, multi-classifier fusion is an important research area because one classifier is unable to perform better than other classifiers on all types of data. In [123], the authors developed gradient histograms using orientation tensors for human actions. A classifier fusion based framework using statistical fusion such as GMM fusion and ANN fusion is proposed in [58]. While conflict results can be generated by different classifiers, previous results have indicated that the fusion of multiple different results can improve the performance of individual classifiers. In general, classifier ensemble is a good resolution to conflict classification results. However, how to find a good way to fuse these classification results from different classifiers remains as a big issue.

Meanwhile, how to utilize multiple features [47–49,62,68,109,124–126] by different feature extraction models from multimedia datasets is another hot research area in the past decade. Different classification models can be employed for different kinds of features, which may discover different properties of the data [61]. The authors in [59] found the complementary nature of the descriptors from different viewpoints, such as semantics, temporal, and spatial resolution. They also employed a hierarchical fusion

that combines static descriptors and dynamic descriptors. Since high-level semantics are sometimes difficult to be captured by visual features, textual features were used in [127] and a sparse linear fusion scheme was proposed in their work to combine visual and textual features for semantic concept retrieval and data classification. Generally speaking, the existing work on multi-classifier ensemble models falls into four types as follows.

2.5.1 Weighted Combination Strategy

The weighted combination strategy is a popular and straight-forward strategy and commonly used in many classifier ensemble models. Two examples are sum and product approaches as the weighted combination rules. The sum rule treats the sum value as the arithmetic mean; while the product rule treats the product value as the geometric mean. The sum rule is equivalent to the product rule for small deviations in the classifier outcomes under the same assumption [128]. In general, the product rule is good when the individual classifiers are independent.

Furthermore, the sum and product rules can be generalized to the weighted combination approaches for different scores. The key to this strategy is to find a suitable set of weights for different scores generated by different classifiers. Several different strategies were proposed in the past in order to determine the weights. For instance, an information gain method for assigning weights is explained in [129] where the authors adapted and evaluated several existing combining methods for the traffic anomaly detection problem and showed that the accuracies of these detectors could be improved. Meng et al. [130] utilized the normalized accuracy to compute the weights for each model built on a specific image patch. In a recent study proposed

in [131], the researchers further extended this method by first sorting all the models according to the interpolated average precision and then selecting the models with top performance. The number of models to retain in the final list is determined via an empirical study.

Although several experimental results indicate that sometimes a weighted combination strategy can give a relatively good performance, the success of this kind of approaches relies on specific knowledge from domain experts or experience from data mining researchers to provide a good estimation of weights. This clearly shows the importance of proper choice of weights.

2.5.2 Statistics Based Strategy

The sum rule can be also considered as a special statistics-based strategy. Some other commonly used approaches in statistics-based approaches are "sum", "max", "min", and "median" rules. The "sum" strategy here is somewhat different from the sum rule in the weight combination strategy and gives an estimation of the final score based on the majority-voting theory. By setting all the weights to the same value, it is then equivalent to the sum rule. The "max" fusion approach is a relatively conservative estimation, where the highest score of all the models is chosen. On the other hand, the "min" fusion strategy picks the lowest value. The fourth one, "median" rule, gets the median value of all the scores.

Kittler et al. [132] developed a common framework for classifier combination and showed that many existing schemes could be considered as special cases of compound classification where all the features are used jointly to make a decision. In [133], Kuncheva evaluated the advantages and disadvantages of these strategies in details

from a theoretical point of view. The main advantage of the statistics-based approach is the low time complexity, while the main disadvantage is that the performance of these models is not quite stable under the condition that the underlying models are not accurate.

2.5.3 Regression Based Strategy

The regression-based strategy receives a lot of attentions recently. In this research direction, the logistic regression based model is commonly utilized. Parameters are estimated using the gradient descent approach in the training stage. After the parameters are learned, the score of a testing data instance can be computed. In [134], a novel logistic regression model is trained to integrate the scores from testing data by different classification models to get a final probabilistic score for the target concept.

Although the logistic regression model sometimes gives relatively robust performances in practice, the disadvantage of regression-based strategy is that this algorithm may suffering from the problem of overfitting.

2.5.4 Bayesian Probabilistic Strategy

With the assumption of the scores are conditionally independent with each other, the Bayesian theory is also widely used in multi-classifier fusion, and sometimes is combined with other strategies [135]. The main issue of this strategy is that the previous assumption does not hold under most circumstances. The final score is computed using the Bayesian rule based on all the scores from the models.

The theory of Dempster-Shafer is an improved method of the Bayesian theory for a subjective probability. It is also a powerful method for combining measures of

evidence from different classifiers. The authors in [136] developed another classifier combination technique based on the Dempster-Shafer theory of evidence by adjusting the evidence of different classifiers based on minimizing the Mean Square Error (MSE) of training data. However, this kind of approaches may still give a relatively bad performance because of the severe deviation from the independence assumption in real cases.

CHAPTER 3

Efficient Deep Learning Based Imbalanced Multimedia Concept Retrieval

In this chapter, the proposed imbalanced multimedia concept retrieval framework is introduced in Section 3.1. The top most challenge that deep learning faces today is to train the massive datasets available at hand. As the datasets become bigger and more complex, deep learning is in its path to be a critical tool to cater big data analysis. Apparently, the most non-trivial tasks in training a deep neural network is the training portion [76]. Since the iterative processing in Deep Learning is integral to the accuracy, it is often very problematic to parallelize these recursive trainings [137]. Thus, the Spark implementation of our proposed framework is presented in Section 3.2 with its performance evaluated using the experimental results in the next section. The last section concludes the findings.

3.1 Framework

3.1.1 CNN Structures

As to be introduced in this section, the deep learning frameworks like CNN have been proven to be one of the most significant developments recently and is famous for

its ability to create multiple levels of training and abstraction that help to understand the data easily. This section discusses how the CNN framework can be utilized for the multimedia imbalanced datasets.

CNN is a subdivision of the deep neural network chain in deep learning that, at the root, is the variants of multilayer perceptron. CNNs are configured to utilize minimum resources in preprocessing [49]. This is done with two techniques: the first is to limit the links among the invisible sections and the input section so that each invisible section links to only a subset of the input called feature maps. The motivation behind the technique of having locally linked networks is taken from the visual cortex where neurons also have local receptive fields [138]. The second technique is to develop simplified computations of images. Since natural images have the tendency of being stationary, the statistics of the different regions of natural images are similar. The second technique takes the advantage of this, utilizes a random subset of trained features, and convolves them to acquire feature activations of the remaining image. Then these acquired features can be used either directly or as ensemble statistics for classifying the data. The ensemble statistics have the characteristics of being comparably very low dimensionality and not overfitting the model as well.

Generally speaking, a CNN model consists of three kinds of layers, which are convolutional, pooling, and fully connected layers [139]. The convolutional layer is composed using several feature maps as defined in Equation (3.1). The feature map of the l^{th} layer and j^{th} feature batch X_j^l is evaluated by convolving the feature maps of its preceding layer X_j^{l-1}. The convolution process uses the activation function f with trained kernels K_{ij}^l and an additive bias b_j^l. The first layer X_j^1 corresponds to the input data, a logistic function is selected as the activation function f that corresponds

an assortment of the input maps.

$$X_j^l = f\left(\sum_{i \in M_j} X_i^{l-1} * K_{ij}^l + b_j^l\right), l \geq 2 \tag{3.1}$$

Here, a feature map is divided into several batches M_j, where M_j represents the data batches and i is the index of each those batches. The pooling layer takes in the input features as given in Equation (3.2) and outputs a subsampled version of it. Here, the operation "pool" stands for a pooling procedure that evaluates the ensemble statistics of the input maps, β_j^l depicts the multiplicative bias, and b_j^l shows the additive bias. The pooling layer is normally placed after each convolutional layer and it typically is designated as a mean or max pooling procedure.

$$X_j^l = f\left(\beta_j^l pool\left(X_j^{l-1}\right) + b_j^l\right), l \geq 2 \tag{3.2}$$

The fully-connected layer is developed to be the high-level reasoning layer in the network. It is placed after or close to the terminal layers of the neural network. All neurons from the earlier layers are then connected to each neuron in the fully-connected layer.

3.1.2 CNN with Bootstrapping

The proposed deep learning framework is contrasting from the negative bootstrap framework developed in [140] that joins random sampling and adaptive selection to recursively search the related negatives. The proposed bootstrapping sampling technique integrates oversampling with decision fusion to improve the CNN's classification accuracy on multimedia data that may or may not have skewed class attributions.

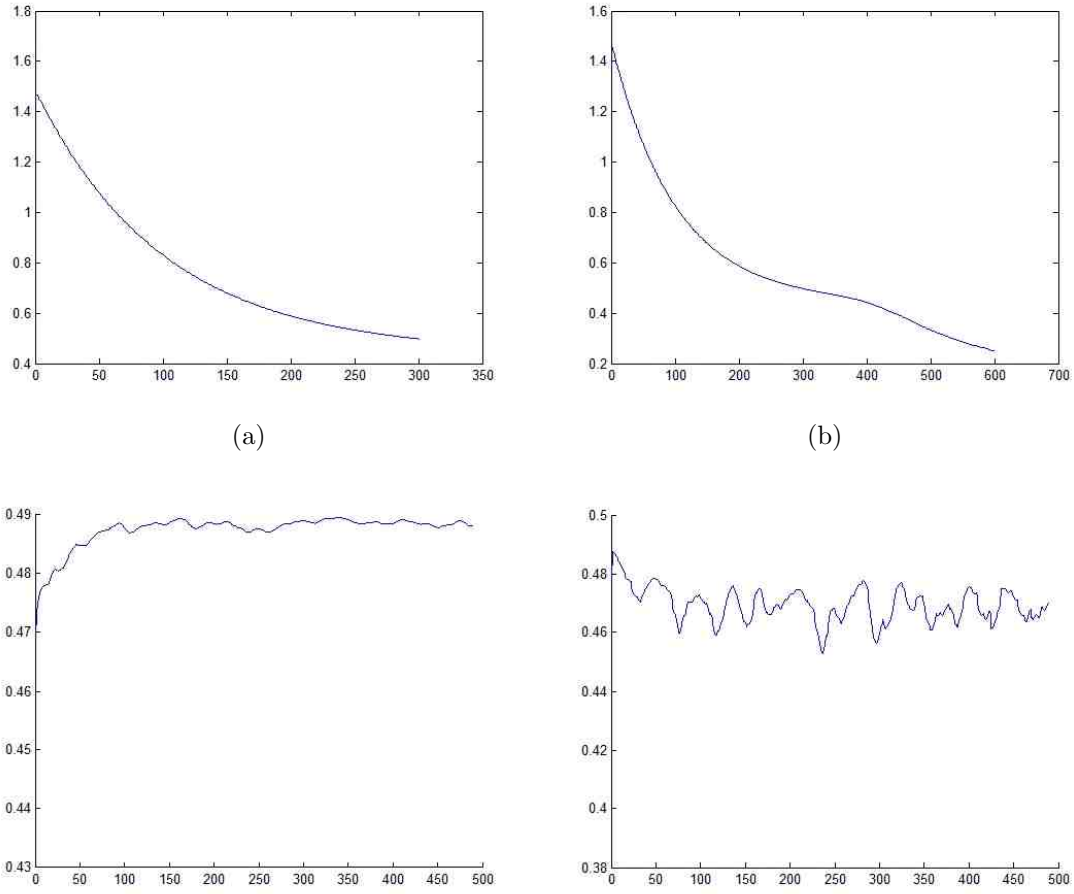
Even after the tremendous success of deep learning frameworks, to the best of the author's knowledge, only a handful of papers target the challenging problem of skewed

class multimedia data. As a matter of fact, directly applying deep learning methods on imbalanced data ends in a very bad classification accuracy. This is illustrated by the empirical study results comparing the balanced and imbalanced datasets in Figure 3.1 where x-axis is for the number of training iterations while y-axis is for error rate, and the error rate of the prediction is compared to the increasing number of iterations of a CNN deep learning network. In the case of balanced datasets in Figures 3.1(a) and 3.1(b), the error rates steadily decrease to definite points. However, when the CNN is used with imbalanced datasets as shown in Figures 3.1(c) and 3.1(d), the prediction error rates waver about plateau points. The reason behind this oscillating error rate is because the deep learning training stages allot the training data into groups. The grouping becomes unfair when the data is imbalanced and some groups may end up containing all negative data instances and no positive data instances. The result of this imbalanced class distribution is a low accuracy classification model.

Conventional bootstrapped samples have the imbalanced dataset with $n >> m$, where n and m are the numbers of negative and positive data instances, respectively. The proposed framework generates batches of s_{neg} and s_{pos} data instances to balance out the ratio $r = s_{neg}/s_{pos}$, where s_{neg} is the number of negative data instances and s_{pos} the number of positive ones. Therefore, totally M batches will be generated, where:

$$M = \frac{n}{s_{neg}} \tag{3.3}$$

Another way to see it is when n is not exactly divisible by s_{neg}, any remaining negative data instances will be removed in the training stage. Since the total count of negative data instances n in the training set is large and the batch size $s = s_{neg} + s_{pos}$ is typically small (i.e., resulting in a small s_{neg}), the removed negative data instances



(a) (b)

(c) (d)

Figure 3.1: Difference in the total error rate produced from (a) & (b) balanced datasets and (c) & (d) imbalanced datasets

become negligible comparative to the data instances used in the training stage. Then from the m positive data instances, one positive data instance is randomly chosen s_{pos} times and combined with s_{neg} negative data instances for each batch. This process is repeated I times to produce the batches in each learning repetition until the error rate converges. This random stochastic process assures an equivalent probability for each positive data instance to be selected and trained with various negative data instances and eventually avoid overfitting. Algorithm 1 illustrates the discussed process. In each repetition process, the bootstrapping process produces a pseudo balanced training set from the original imbalanced dataset, which can be then used by the deep learning model for learning.

Algorithm 1 Proposed module of CNN with bootstrapping

```
Split the training set into a positive set pos and a negative set neg
Divide neg into M batches, each with s_neg negative data instances
for 1 to I do
    for 1 to M do
        for 1:s_pos do
            randomly pick one data instance from pos;
        end for
        combine the data instances in pos and neg together;
    end for
    Train a CNN model;
end for
end;
```

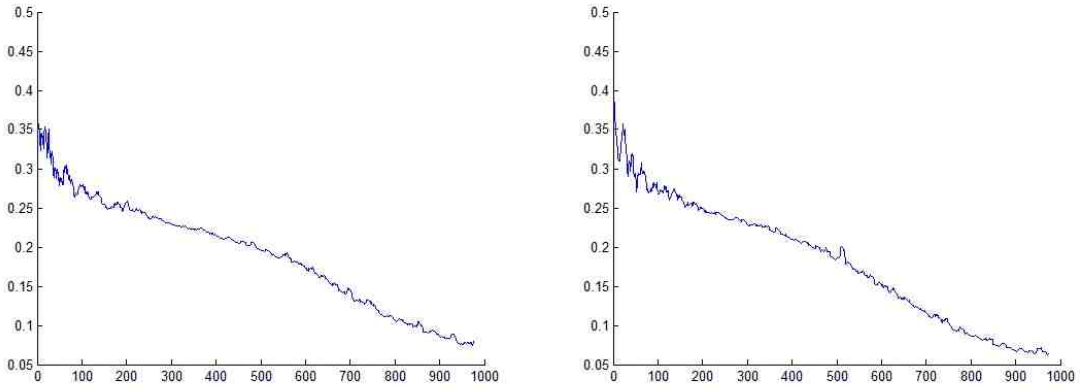
Let the size of each input be $m \times m$ such that a four mid-layer CNN forms as depicted in Table 3.1, where k_L represents the number of mask neurons applicable on a given subgroup of input values and $n_L \times n_L$ indicates the size of each mask in the L_{th} convolution layer. The output from the L_{th} convolution layer is given to the L_{th} pooling layer and it is split into a group of non-overlapping rectangles of size $p_L \times p_L$, where the pooling operations are applied for downsampling. Furthermore, the

bootstrapping method explained earlier is then used to generate N batches of balanced training instances that are given to the first layer of CNN in iterations. The input layer is followed by two convolutional layers, and then followed by their respective mean pooling layers. The first convolutional layer produces the inner product of the $k_1 \times (n_1 \times n_1)$ masks and passes the output to the first mean pooling layer. Mean pooling layers summarize the outputs of the neighboring subsets of masks in the same kernel map. The output of the pooling layer is passed as the input to the second convolutional layer. This is followed by the mean pooling layer using the same process as mentioned earlier but with different mask sizes. The size of the vector of the final CNN layer denotes the number of classes attributed to the data. The experiment performs binary classification, and thus the size is set to 2.

Table 3.1: Training parameters for CNN

Layer	Layer size	Output size
Input ($m \times m$)		
Convolution 1	$k_1 \times n_1 \times n_1$	$k_1 \times (m - n_1 + 1) \times (m - n_1 + 1)$
Pooling 1	$p_1 \times p_1$	$k_1 \times (m - n_1 + 1)/p_1 \times (m - n_1 + 1)/p_1$
let $m_2 = (m - n_1 + 1)/p_1$		
Convolution 2	$k_2 \times n_2 \times n_2$	$k_2 \times (m_2 - n_2 + 1) \times (m_2 - n_2 + 1)$
Pooling 2	$p_2 \times p_2$	$k_2 \times (m_2 - n_2 + 1)/p_2 \times (m_2 - n_2 + 1)/p_2$
Output		2

Finally, the consequent convolution masks and weights are used for classification and the class receiving the highest score will be attributed to the test data instance. The benefit of the bootstrapping method can be observed from the prediction error rates shown in Figure 3.2 as it is applied to the imbalanced dataset of Figures 3.2(a) and 3.2(b), respectively. The descending error rates illustrate the effectiveness in the convergent process.



(a) (b)

Figure 3.2: Total error rates convergence generated from imbalanced datasets (a) & (b) using the proposed bootstrapping method

3.1.3 Integration of CNN and Low-level Features

The training time of CNN is notorious for being computationally taxing. For example, it took one month to train 1755 videos to achieve sufficient performance metrics [43]. It is observed in the literature that training a deep learning method is substantially longer if provided with the raw data at the input layer. To improve the efficiency of CNN, we first propose a different approach by using the low-level multimedia features that are much smaller than the raw signal data. This key process here is that we propose to feed the low-level features into the CNN directly to substantially reduce the m value (18 in the proposed experiment) and greatly improve the processing times. These low-level features are composed of Haar [141], HOG [72], HSV, YCbCr [142], and CEDD [143]. These are chained into a feature vector which is then transformed to a matrix using PCA (Principle Component Analysis). This transformation is required because CNN does not support one-dimensional vectors and the features are fed as an 18 by 18 matrix. The sizes of the matrix and masks in

Table 3 are decided based on empirical studies and could be adjusted with different feature dimensions and datasets.

The internal deep learning process of the CNN is similar to what is described in the previous section, except for the pooling layers that are removed because the low-level features are not necessarily stationary in every iteration. Table 3.2 illustrates the detailed training parameters used in the proposed framework. Since we have earlier reduced the dimensions of the input features, a relatively small mask size can be applied, in comparison to that in CaffeOnSpark website [144].

Table 3.2: Training parameters in the proposed framework

Layer	Mask size	Output size
Input (18×18)		
Convolution 1	$6 \times 3 \times 3$	$6 \times 16 \times 16$
Convolution 2	$9 \times 3 \times 3$	$9 \times 14 \times 14$
Output		2

3.2 Deployment of the Proposed Framework on a Spark Cluster

A shortcoming of using low-level features is potentially losing information. Thus, another idea of making an efficient framework is to build it on top of a big data processing system. In this chapter, we built the proposed model on top of the dedicated Spark cluster. The cluster is running most recent versions of the required distributed big data infrastructure, i.e., Apache Hadoop 2.7.3 with Yarn and Apache Spark 2.0. The developed Spark cluster serves as the primary test bed cluster for deep learning and distributed processing experiments [47–49, 109, 124].

Figure 3.3 illustrates the core infrastructure of the cluster. There are 4 nodes in total with the master node connected to a dedicated class-c dedicated IP. The overall cluster configuration is heterogeneous but is normalized to the least performing node in the infrastructure. Each node is setup to instantiate 2 workers with 4 GB of memory and 1 TB of storage. The data files were replicated across the three Hadoop HDFS instances in all data nodes, namely data node 1, data node 2, and data node 3 as shown in Figure 3.3.

Figure 3.3: The infrastructure of the built spark cluster

The main abstraction that Spark provides is a resilient distributed dataset (RDD), an in-memory storage abstraction of frame confidence score elements partitioned across the nodes of the cluster that can be operated on in parallel. These RDDs were created from confidence score files present in Hadoop Distributed File System (HDFS). The proposed CNN framework ran on Spark as tasks coordinated by the master node in the cluster which has a driver program. Linear speedup was achieved

by modifying the Spark configuration and setting parallelism to the number of cores present in the cluster to utilize the cluster to its maximum capacity. It is recommended that a maximum of 2 to 3 tasks are to be run on a CPU core at an instance of time. The number of tasks executed in parallel on a node is equal to the number of cores in the corresponding node. Spark automatically sets the number of map tasks to run on each file according to the number of partitions present. A partition is defined by each file which is loaded from the HDFS. Executors were started on each node of the cluster to perform the tasks.

Deep learning methods are notorious for computationally expensive and impractical for streaming data. The attempt to overcome this challenge is to use a distributed environment and Spark. By the empirical testing and evaluation, it was observed that the same neural network implementation using Java in Apache Spark 2.0 achieved 400% speed improvement over Matlab 10 performed on the same cluster.

There are a lot of use-cases where multi-core or GPU based processing or conventional HPC systems may be significantly faster than any Spark implementation. We have to take all the feasibility cases into account and argue the case of building a system to make positive forward progress in this research. Since Spark is only good with recursive statements and streaming inputs, in the case of classifying data, MapReduce will probably do a competitive job to Spark due to the fact that there is only one time read involved from the hard disk.

Although the Spark clusters are proven to be suitable for recursive computing, how to distribute the neural network training remains as a challenging issue. Since all parameters in a certain layer are updated after training each mini batch, most popular neural network models cannot be deployed on distributed computing clusters

to run parallel processing. Another problem to deploy the neural network frameworks on Spark is that Spark supports Scala and Java stably, but only partially supports Python. Hence, we use a deep learning tool written in java called Deeplearning4j [145] to implement parallel computing on training the neural network models.

The basic idea of training a neural network model on Spark is parameter averaging. As drawn in Figure 3.4, the input data is divided into several subsets based on the configuration of the master node. With the same as the regular neural network training steps (please note that the "data split" here is different from the "batch"), the Spark driver (i.e., the master node) starts with a randomly generated parameter set and an initial network configuration. For each subset of the training data, the master node distributes the parameters, configuration, and network updater states to all slave nodes. Then, each slave node trains its own portion of the subset and updates the parameters as well. After several iterations, the parameters and states are sent back to the master node which will average the parameters and states to update the trained neural network. Then, the average values are distributed to the slave nodes as well as the workers again for further training.

In this study, we identify the best available features to identify the skewed and imbalanced classes and improve the classification of the imbalanced datasets. It is an industry best practice to start with the current best features and further improve the performance by processing the skewed and imbalanced classes. Recently, those features extracted from pre-trained deep learning models are proven to outperform traditional low-level features. In this chapter, we use a pre-trained and fine-tuned CNN model on the ImageNet data, Alexnet [82], for keyframe feature extraction. The Alexnet structure is well-trained and proven with great performance. It contains

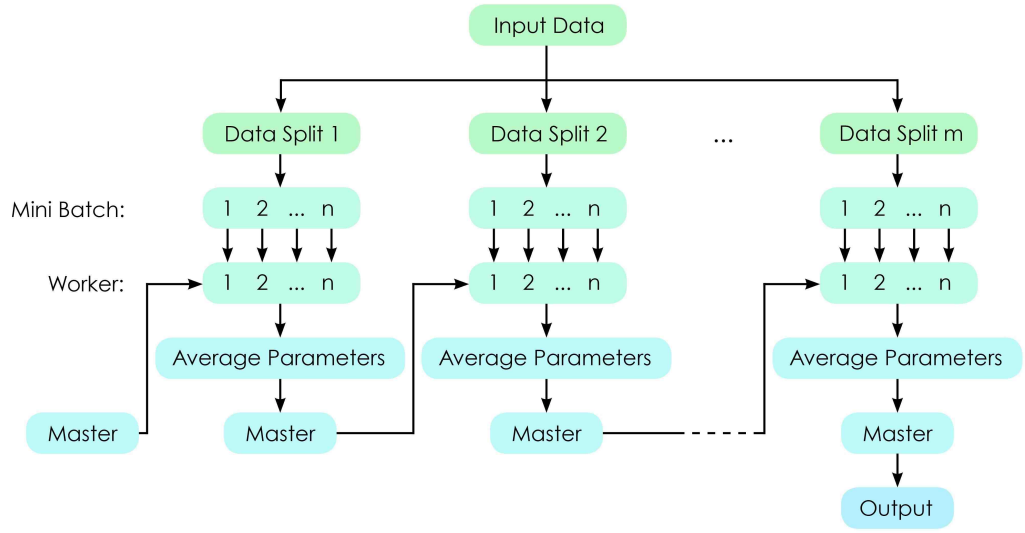


Figure 3.4: The flowchart of training CNNs on the built Spark cluster

five convolutional layers and three fully-connected layers and the CNN features are extracted from all the training and testing keyframes from the output layer, i.e., the 8th layer with one-thousand dimensions. These features are finally fed to a neural network with two fully connection layers, where the first layer contains 100 neurons and the second one is composed of 10 neurons.

3.3 Experimental Results

3.3.1 Evaluation on the UCF11 Dataset

First, the proposed model is tested on a relatively balanced video dataset, UCF YouTube Action (UCF11) dataset [146], to prove the efficiency of feeding the low-level features to the CNN models. UCF11 includes videos collected from YouTube with various problems like non-static background, low video quality, camera motions,

poor illumination conditions, etc. It is a relatively balanced dataset as compared to the TRECVID dataset and contains 11 action categories: basketball shooting, biking/cycling, diving, golf swinging, horseback riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volley ball spiking, and walking with a dog. This dataset is very challenging [147] due to the large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, and etc. For each category, the videos are grouped into 25 groups with more than 4 action clips in it. The video clips in the same group share some common features, such as the same actor, similar background, and similar viewpoint. The STIP features [148] are extracted from each UCF11 video. STIP is built on the idea of the Harris and Forstner interest point operators to detect local structures in space-time where the image values have significant local variations in both space and time. STIP features can be obtained by estimating the spatio-temporal extents of the detected events and computing their scale-invariant spatio-temporal descriptors. The dimensions of the STIP features are reduced to 32 from 162 by applying the principle component analysis (PCA) technique for fast computation purposes. In the video representation part, 256 Gaussian components in GMMs (Gaussian Mixture Models) are used and the leave-one-out cross validation scheme is employed. Since UCF11 is a relatively balanced dataset, in the bootstrapping step, a small number of data instances are randomly picked from each category (5 in thi experiment) to form the batches for CNN training. Figure 3.5 shows some example frames from the UCF11 dataset.

Table 3.3 shows the confusion matrix of applying the proposed framework to the UCF11 dataset. Here, “Bas” denotes basketball shooting, “Bik” is for biking/cycling,

Figure 3.5: Example of UCF11 dataset with approximately 1168 videos in 11 categories

and so on. The vertical labels are the ground truth, i.e., the actual labels; while the horizontal side shows the prediction labels. The number in each grid shows the percentage of the data instances. For instance, the number "85" shows that 85 percent of the 'horseback riding' testing instances are correctly identified; while the number "1" shows that 1 percent of the horseback riding data instances are misclassified as soccer juggling.

Table 3.3: Confusion matrix of the UCF11 Dataset

Truth \ Prediction	Bas	Bik	Div	Gol	Hor	Soc	Swi	Ten	Tra	Vol	Wal
Basketball	**55**	5	3	8	1	1	2	13	0	11	1
Biking	1	**73**	0	0	10	0	3	3	2	2	5
Diving	5	2	**76**	1	1	1	2	1	1	6	3
Golf Swing	12	1	1	**82**	0	1	2	2	0	0	0
Horse Riding	1	6	1	0	**85**	1	1	1	1	0	6
Soccer Juggling	4	1	1	4	5	**63**	6	5	1	4	5
Swinging	1	4	4	1	1	1	**79**	0	4	3	2
Tennis Swing	8	1	1	8	4	3	2	**72**	1	1	1
Trampoline Jumping	1	0	1	0	2	9	8	1	**77**	1	1
Volleyball Spiking	7	1	2	1	0	2	1	8	0	**79**	0
Walking with a dog	2	7	2	3	20	1	2	5	2	0	**54**

Table 3.4 shows the performance comparison between the proposed approach and three other state-of-the-art methods. Specifically, [123] used the combination of Histograms of Gradients into orientation tensors and applied SVM as the classifier. In [147], motion features based on the ROI (Region of Interest) estimation and AdaBoost were used to integrate all the heterogeneous yet complementary features for recognition. In [149], SVM was applied to a tensor motion descriptor with optical flow for action recognition. As shown in Table 3.4, the proposed approach achieves the best accuracy rate among all the methods. This experiment clearly proves that while this framework aims to address the challenges caused by a highly imbalanced data distribution, it is also very effective in classifying relatively balanced datasets.

Table 3.4: Result comparison for the UCF11 dataset

Group	Accuracy
Perez et al. [123]	68.9%
Liu, Luo, & Shah [147]	71.2%
Mota et al. [149]	72.7%
The proposed framework	**72.8**%

3.3.2 Experimental Results on the TRECVID Dataset

In order to demonstrate the effectiveness of the proposed framework for imbalanced multimedia data classification, the TRECVID dataset [150], a large-size benchmark dataset with highly skewed data distribution, is used in the experiment. In particular, the IACC.1 dataset from the TRECVID 2015 datasets (Over et al., 2015) is used. The semantic indexing (SIN) task in TRECVID 2015 aims to recognize the semantic concept contained within a video shot, which can be an essential technology for retrieval, categorization, and other video exploitations. Here, the concepts refer to the high-level semantic objects such as a car, road, and tree. Figure 3.6 shows four sample keyframes with the labelled concepts. There are several challenges such as data imbalance, scalability, and semantic gap. As a result, traditional deep learning approaches, including CNNs, often perform poorly on the TRECVID dataset due to the problem of under-fitting, huge diversity, and noisy and incomplete data annotation [41, 42]. Please note that the data imbalance degrees of different concepts vary in the TRECVID dataset, and thus a fixed batch size may not be suitable for every testing concept. To address this issue, the batch size is chosen dynamically based on the number of positive data instances in the training set. In this experiment, the batch size is set to be twice bigger than the number of positive training data instances. Sample keyframes with annotated concepts in the TRECVID dataset are shown in Figure 3.6.

(a) (b)

(c) (d)

Figure 3.6: Sample keyframes with annotated concepts in the TRECVID dataset: the concepts are (a) face, (b) politics, (c) bicycling, and (d) tree, respectively

Here, the TRECVID dataset is chosen mainly because it contains a large number of data instances and is highly imbalanced [151]. In the selected IACC.1 dataset, a total number of 262,911 data instances are used for training; while 113,046 data instances are used for testing. The proposed framework is evaluated on 84 concepts with severely skewed data distributions and P/N ratios lower than five ten-thousandths, where the P/N ratio is the ratio between the number of positive data instances and the number of negative data instances. As indicated in [152], in imbalanced data classification, the recall metric is considered more important than precision and the F-score represents the trade-off between precision and recall. As shown in Figures 3.7 and 3.8, the recall and F-score values of the proposed framework using the low-level features and the features from the pre-trained CNN models are compared with the scores from TiTech (Tokyo Institute of Technology) that achieved the best performance in the semantic indexing task several times in the past years [153].

In case of the TRECVID confidence score evaluation, we have to work with the unstructured key-value pairs of the TRECVID video shots. There is a need to store the massive TRECVID multimedia data, in the order of several Terabytes, with redundancy over the years since 2003 until recent. We have stored the TRECVID video frames as well as the extracted confidence scores that are continuously used in the models to compare with previous datasets or to train for the recent competition. The photos and video frames can be stored in the HDFS and processed using the Spark engine and the confidence scores can be assigned and stored back in the HDFS. The resultant confidence scores are unstructured key-value pairs that need to be stored in the HDFS based redundant data store and accessed for data mining processing. Therefore, Spark befits as a perfect candidate solution to this problem.

From the results drawn on Figures 3.7 and 3.8, the F-scores generated from the low-level features are higher than those of the TiTech group for two thirds of the 84 concepts; while the results by those features from the pre-trained CNN models perform better than four fifths of the TiTech scores. For the recall measurement, both of the proposed frameworks generate better results for almost every concept. The only exception is when using the low-level features, they may fail to identify a true positive data instance due to the noisy data annotations and information lost in feature extraction. It is also worth noting that for 50 concepts, the TiTech group can only locate zero or one true positive data instance; while the proposed approach reaches more than 0.628 recall value on average. This clearly demonstrates the effectiveness of integrating CNNs with the bootstrapping strategy in the proposed framework for imbalanced multimedia data classification, especially on the fact that the study in [154] showed that the performance of CNNs is far worse than all other classifiers the authors tried on the TRECVID dataset.

3.4 Conclusions

In this chapter, we proposed to extend the CNN-based deep learning technique by incorporating a bootstrapping algorithm. Moreover, to achieve faster computation speeds and better handling of unstructured key-value pairs of the TRECVID video data, we harnessed the power of Apache Spark. The Spark system is implemented on a dedicated Spark cluster developed solely for the computational needs of the research. In the bootstrapping stage, pseudo balanced training batches are rendered and inserted into the CNN for classification. The experimental results establish the effectiveness of the proposed framework for accurately classifying highly imbalanced

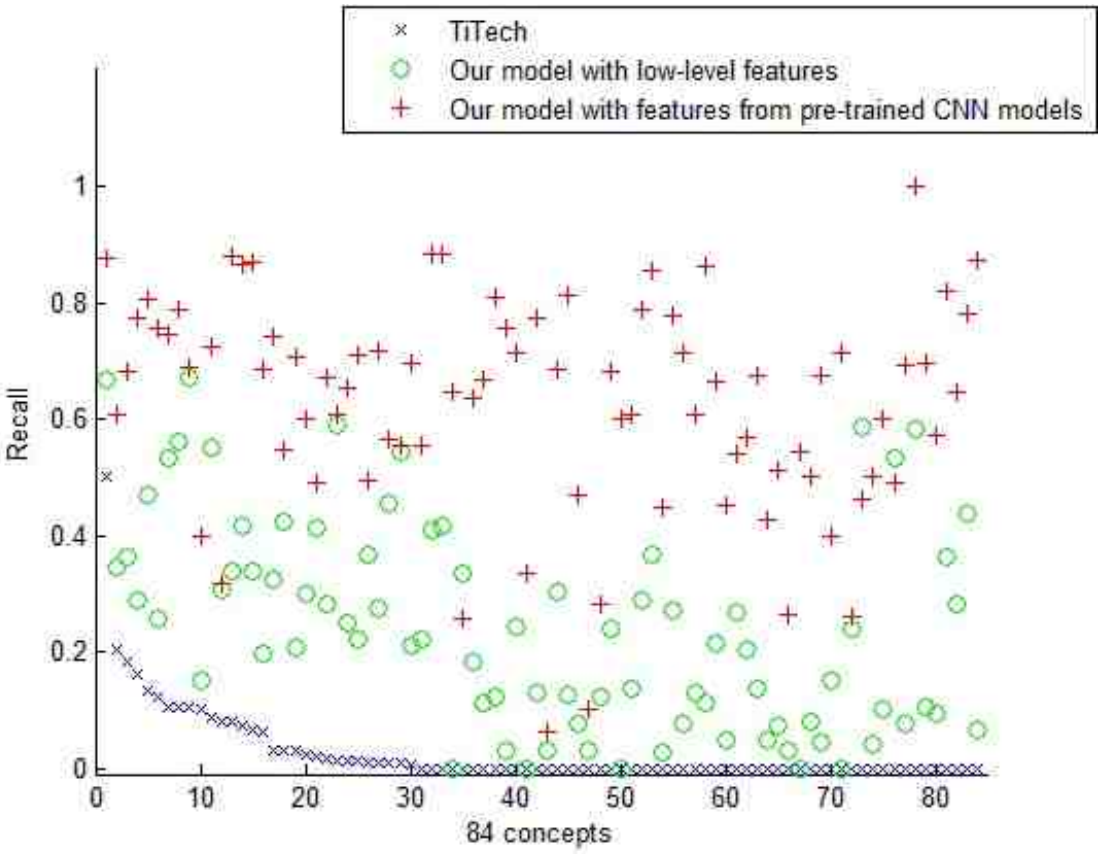


Figure 3.7: Recall comparisons on all imbalanced concepts

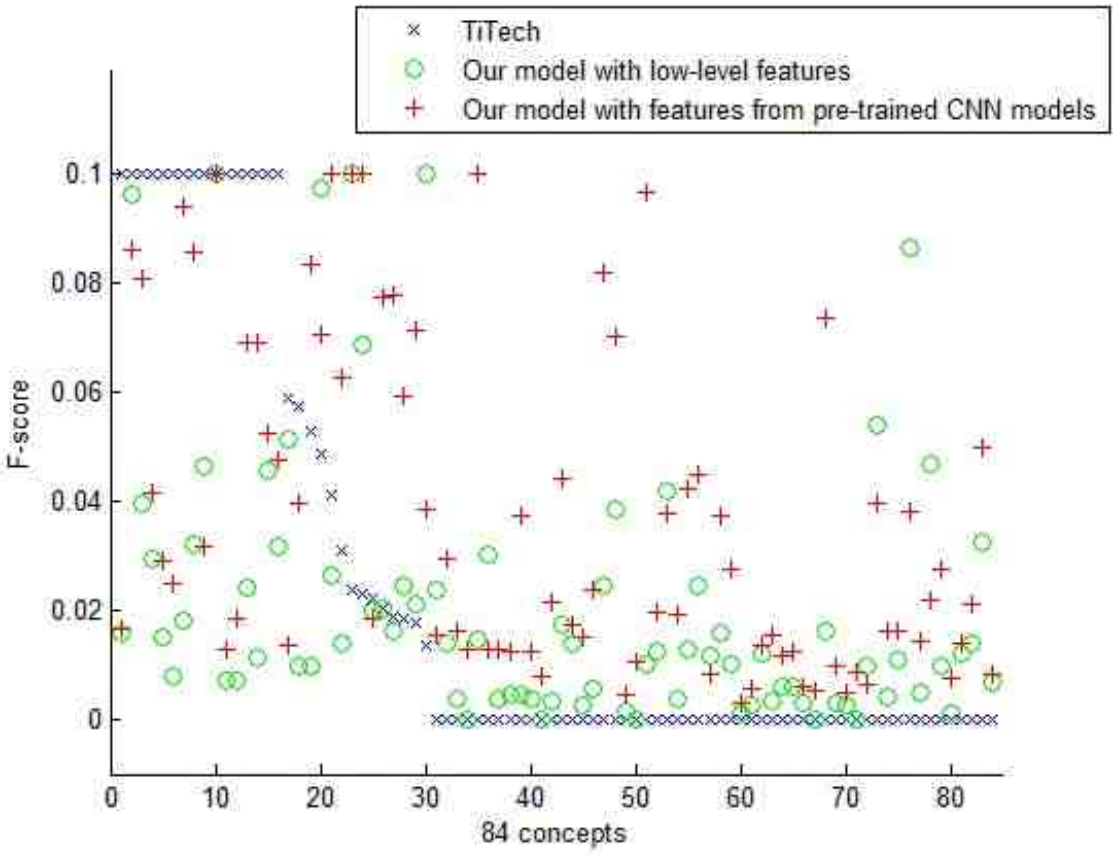


Figure 3.8: F-score comparisons on all imbalanced concepts

multimedia data. Different from many existing methods in deep learning that take the required raw media data in the input layer, the proposed deep learning framework works efficiently on the low-level features, which largely reduces the required training time in deep learning. Furthermore, a computational boost is achieved with the power of distributed computing using Apache Spark and better information retrieval results are generated by the features from the pre-trained CNN models.

Though we propose a powerful imbalanced big data processing system using Spark in this chapter, running deep learning algorithms on GPU is much more efficient than on CPU. Therefore, it is better to extend the system for accelerating deep learning on Spark applications using GPUs. Since GPUs provide both high-computation capabilities and high-memory bandwidth, they can be used to accelerate both computation-intensive and memory-intensive Spark jobs. In the future, we plan to enhance this system and run deep learning applications on distributed GPUs with Spark.

CHAPTER 4

Correlation-Assisted Concept Retrieval and Score Enhancement

A video shot usually contains multiply concepts which are correlated in real-world multimedia datasets, either positively or negatively. In other words, some concepts co-occur more frequently, e.g., sea and whale; while others rarely co-occur, e.g., sky and meeting. Such correlations can provide important context cues to help detect the concepts [44, 46, 92, 155, 156]. This chapter is organized as follows. In the Section 4.1, a hierarchy is built using the inter-concept correlations. The second section describes a novel idea of enhancing imbalanced concept detection using the correlation between the retrieval scores and labels. The third section shows how to setup the framework and compares the results of the proposed system on the TRECVID dataset. Finally, the last section draws the conclusion and identifies future research directions.

4.1 Building Hierarchies for Datasets

4.1.1 Conditional Probability Calculation

Although the inter-concept connection information has been proposed to enhance semantic concept retrieval results, most of them utilize the hierarchical relationship

from the data provider [157] for combining the classes to generate reorganized hierarchies. For instance, if a dataset contains labels "apple", "banana", and "fruit", the data provider may give a note "apple imply fruit". In such case, the data instances with the label "apple" would be automatically added to the label "fruit". However, many other kinds of relationships are not that straightforward and relationships generated manually may lead to biases and not suitable for big datasets.

In this framework, we first build a hierarchical model for all concepts based on conditional probabilities generated from the training set. Here we define C_{parent} as a parent concept and C_{child} as a child concept. Let $P(.)$ be the probability, then C^+_{parent} denotes the positive collection of C_{parent}; whereas C^+_{child} represents the positive collection of C_{child}. If C_{parent} is the parent of C_{child}, the appearance of C_{child} should imply the appearance of C_{parent}. As an example, if a video shot contains the concept "car", it definitely includes the concept "vehicle" as well, unless the ground truth is incorrect. In this example, "car" is a child concept while "vehicle" is a parent concept. The probability of C_{parent} appearing increases if C_{child} appears, and the probability of C_{parent} appearing decreases if C_{child} does not appear. This conditional probability can be computed by Equation (4.1).

$$P(C^+_{parent}|C^+_{child}) = \frac{P(C^+_{parent} \text{ and } C^+_{child})}{P(C^+_{child})}. \tag{4.1}$$

4.1.2 Bottom-up Organization

In real-world, some concept pairs have the parent-child relationship (like "sky" and "sun"). This kind of inter-concept relationships should also be considered. In addition, since the concept labels in multimedia datasets are usually manually decided by the volunteers or by some automatic labeling techniques, the ground truth is not

always correct. Therefore, a threshold of 0.9 is set to determine whether two concepts have the parent/child relationship, which is represented as $P(C^+_{parent}|C^+_{child}) > 0.9$.

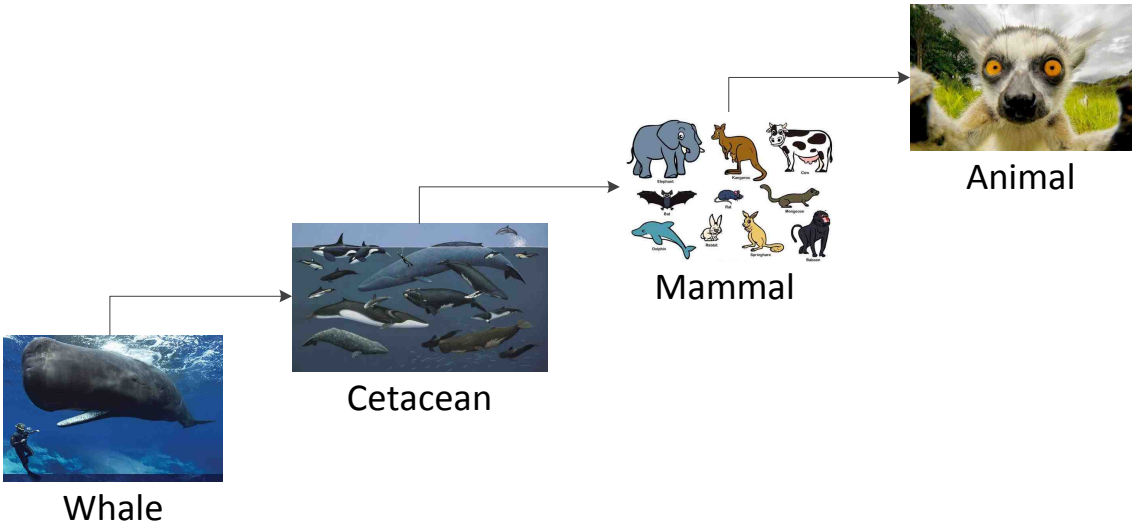


Figure 4.1: Parent/Child relationship examples 1

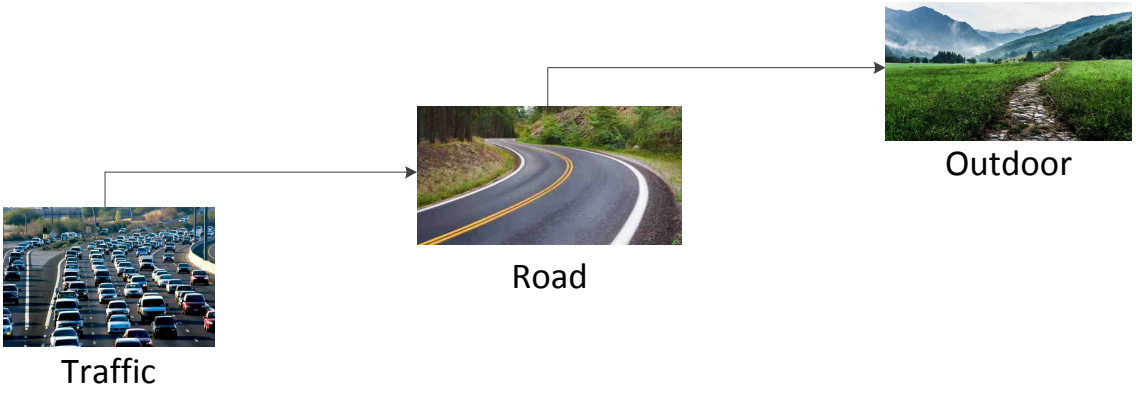


Figure 4.2: Parent/Child relationship examples 2

Next, the hierarchy model of all concepts is built from the leaf nodes (in a bottom-up manner) using all the parent-child concept pairs generated and filtered. If a concept has no child but at least one parent, it is considered as a leaf node and is added to the initial model. The following step shows the example of including the "direct" parent nodes for the "whale" leaf node. A whale is a cetacean, a mammal,

and an animal as well. With the fact that the appearance of a whale implies the appearance of a cetacean and "cetacean" implies "animal", these two concept pairs also have the parent-child relationship. Thus, "whale" is first included as a child node and then followed by "cetacean", "mammal" , and "animal". If a parent concept has no parent like "animal" in this case, it will be finally considered as a root (head) node. These operations are shown in Figure 4.1 and Figure 4.2.

Figure 4.3: Siblings relationship examples 1

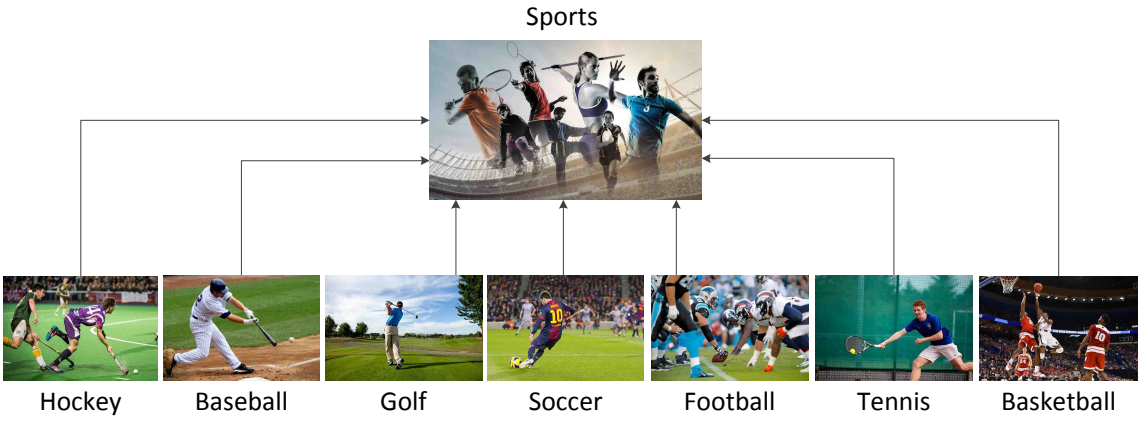


Figure 4.4: Siblings relationship examples 2

After finding out all the qualified parent-child concept pairs, we can combine the branches into a tree and thus find the siblings of the child concepts as given in Figure 4.3 and Figure 4.4. Different tree structures would be generated from different datasets, even from different subsets of a dataset. In the aforementioned example, if the concept "mammal" is removed, "animal" could be the direct parent of "cetacean" in the updated hierarchy. In general, the more concepts included, the more complete the model would be. Though the hierarchy model can never be perfect, it is suitable for the particular dataset on which it based.

4.2 Prediction Score Enhancement for Rare Concept Retrieval

4.2.1 Score Based Correlation Generation

As mentioned in related work, most previous research including the aforementioned conditional probability approaches calculates the inter-concept correlations and builds the hierarchical structures using the label information in the training data, i.e., the appearance or non-appearance of the concepts. One main problem of using such information to leverage the retrieval scores is the correlation coefficients among rare concepts, and correlation coefficients between imbalanced concepts and balanced concepts are usually weak. Suppose that we calculate the correlation between a common concept with 10,000 instances and a rare concept with 10 instances, the correlation coefficient will be small and even one wrong label for the rare concept in the training set will lead to a big mistake in inter-concept correlation calculation and thus cause wrong results.

Another issue is that high correlations between concepts do not necessarily lead to the high correlation between the concepts and the detection (prediction) scores, especially for rare concepts since the quality of scores from imbalanced concepts is often worse than those from the balanced concepts. This is caused by the nature of the original dataset (with a skewed distribution) and directly using rare concepts' correlation information for score integration may even downgrade the original results. For instance, the concept "hurricane" should have a positive correlation with the concept "disaster". However, with the bad prediction scores, the concept "hurricane" does not really help the retrieval of the concept "disaster" in the imbalanced dataset.

There are only 6 out of the total of 137,272 video shots that include the concept "cow" in the TRECVID dataset. This raises the third issue: the detection scores of rare concepts themselves can be relatively imprecise. Most of the classifiers cannot get acceptable prediction scores for these rare concepts albeit with such a big training dataset. To solve these three issues, we propose a model to integrate the prediction scores of the rare concepts using the Pearson correlation coefficients from both the label and score information for score enhancement in this framework.

The Pearson product-moment correlation coefficient [110], denoted by ρ (or r), measures the strength of a linear association between two variables X and Y, and is widely used as a measure of the degree of the linear dependence between X and Y. It attempts to draw a line of best fit through the data of two variables, and ρ (or r) indicates how far away all these data points are to this line of best fit. The ρ (or r) values are between +1 and -1 (inclusive), where +1 is a total positive correlation, 0 is no correlation, and -1 is a total negative correlation. Let $cov(X, Y)$ and E be the covariance and expectation of X and Y, σ_X and σ_Y be the standard deviations of X

and Y, and μ_X and μ_Y be the mean values of X and Y. For a population, we have the following:

$$\begin{aligned}
\rho_{X,Y} &= \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}; \text{ where} \qquad (4.2)\\
cov(X,Y) &= E\{[x-E(x)][(y-E(y)]\} = E(xy) - E(x)E(y);\\
\sigma_X &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu_X)^2};\\
\sigma_Y &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-\mu_Y)^2};\\
\mu_X &= \frac{1}{N}\sum_{i=1}^{N} x_i;\\
\mu_Y &= \frac{1}{N}\sum_{i=1}^{N} y_i.
\end{aligned}$$

Here we define C_T as the label information of an imbalanced (target) concept, and let S_R be the prediction score of a support (related) concept which can be either a balanced concept or imbalanced one. Take "cow" as a target concept. In order to enhance the prediction score of the rare concept "cow", all $\rho_{(C_T,S_R)}$ are calculated and ranked. In the equation, T is the concept "cow" and $R = 1, 2, \cdots, N$ and N is the number of concepts. The top ten related concepts are shown in Figure 4.5, which means the prediction scores of these concepts are helpful to enhance the prediction score of the concept "cow".

As shown in Figure 4.5, the top ten related concepts are "Herbivore", "Ruminant", "Mammal", "Quadruped", "Wild Animal", "Vertebrate", "Animal", "Animal Pens And Cages", "Sea Mammal", and "Cattle", respectively. Clearly, most of them are reasonable at the first glance, expect "Sea Mammal". Nevertheless, the shapes of some sea mammals are similar to those of the cows. Especially, one common kind of

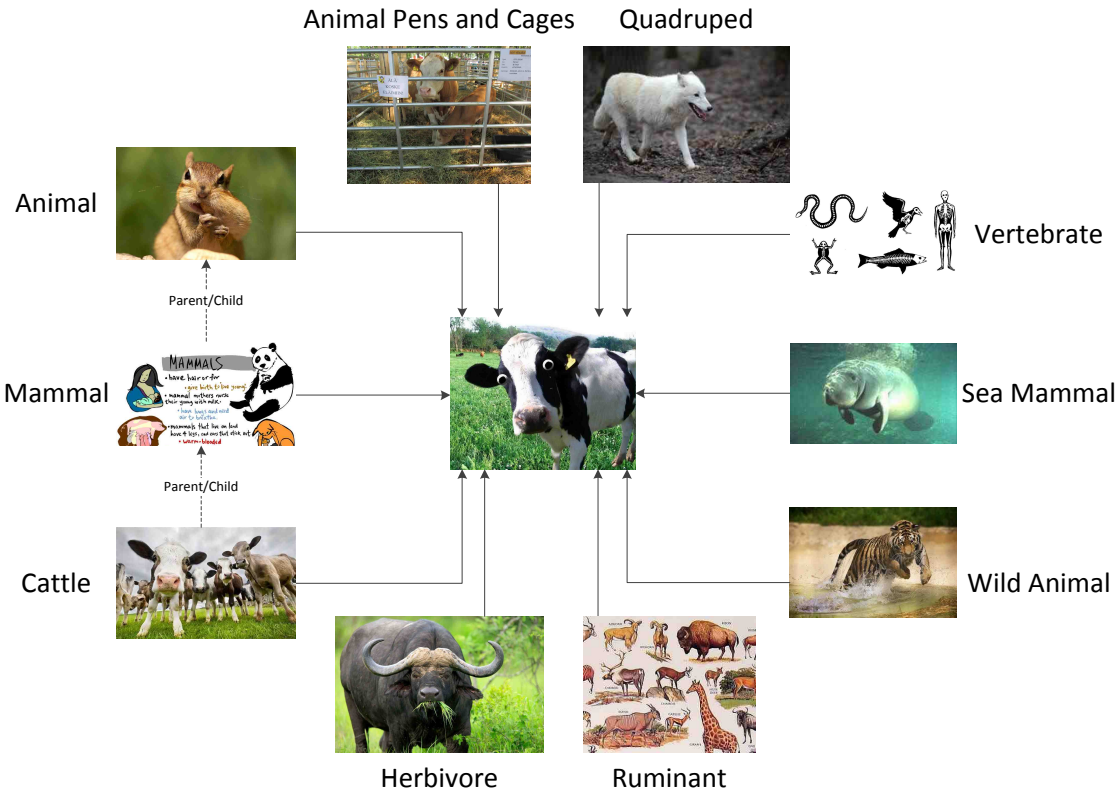


Figure 4.5: Top ten related concepts that support the rare concept "cow"

sea mammal, manatee, is also known as "sea cow". This highlights another advantage of the proposed framework, which can find the potentially related concepts. Figure 4.5 also implies that the prediction score of the concept "cow" itself is imprecise and thus will not be integrated for the enhancement.

4.2.2 Negative Related Concepts

After ranking top ten related concepts by their correlation values and building a hierarchy as shown in Figure 4.5, it can be further expanded using the hierarchical models built in Section 4.1. For a target concept C_T, if a related concept C_R is connected to it, its parent will also be added to the model. In this example, since "Quadruped" is connected to "cow", "Animal" would be included as well. However, since "Animal" is already included based on the ranked scores, we don't need to add it again as shown in Figure 4.5. In this framework, C_T is a rare concept. Afterward, the scores of the top ten related concepts are used to train an integration model using a discriminant analysis classifier.

As discussed earlier, some concepts such as "sky" and "shark" rarely co-occur, which can also provide important context cues to help detect the concepts. Take the aforementioned example, the top ten concepts that have a negative relationship with "cow" are "Hospital", "Bomber Bombing", "Fear", "Factory", "Sports Car", "Disgust", "Handshaking", "Airplane Landing", "Black Frame", and "Network Logo". Here, these 10 concepts are not simply irrelevant to "cow" as they look like, but they also have relatively strong negative relationships. That is, if "Hospital" appears in a testing frame, "cow" is very unlikely to appear in the same frame. Therefore, the

opposite numbers of scores from those negative-related concepts can be integrated in the enhancement framework.

4.2.3 Score Integration

To train the integration model, 5 different kinds of popular algorithms are used, including Support Vector Machine (SVM) [158,159], Naive Bayes (NB) [160], Random Forest (RF) [161, 162], Logistic Regression (LR) [134, 163, 164], and Discriminant Analysis Classifier (DAC) [134, 163, 164].

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane to classify the dataset so that the geometric margin is maximized. A "Naive Bayes" (NB) is a classification technique based on the Bayes' Theorem with an assumption of independence among predictors. A Random Forest (RF) is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Logistic Regression (LR) is another predictive analysis and a kind of generalized linear model. It can be used to conduct when the dependent variable is binary just like in this case. Instead of just predicting binary-valued labels in linear regression, logistic regression uses a different hypothesis class to predict the probability that a given example belongs to the positive (e.g., fraud) class versus the probability that it belongs to the negative (e.g., non-fraud) class by a logistic function. Discriminant Analysis Classifier (DAC) assumes that the data from different classes are generated based on different Gaussian distributions. In the training phase, the fitting function calculates the parameters of a Gaussian distribution for each class;

while in the testing stage, the trained classifier finds the class with the smallest misclassification cost.

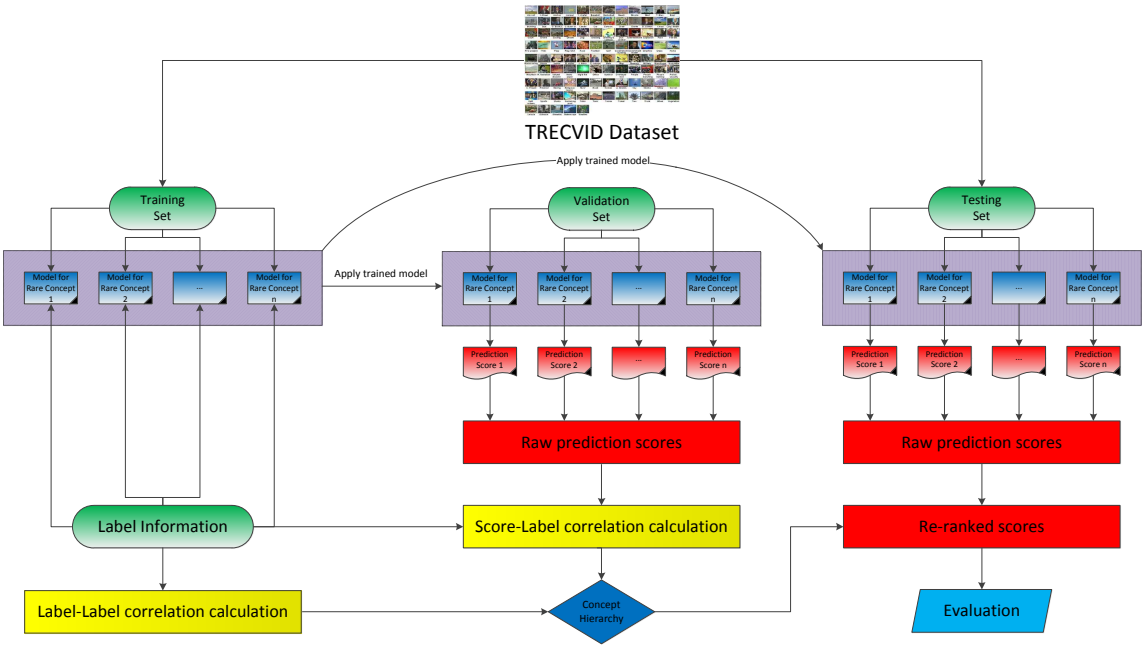


Figure 4.6: The proposed framework

4.2.4 Workflow

The proposed framework includes a training stage as well as a testing stage as shown in Figure 4.6. The testing dataset is first split into three parts, including a training set, a validation set, and a testing set. In the training phase, the training set conditional probabilities are calculated to build a hierarchical model for all concepts from the training label information.

Next, for all the validation video shots and N concepts, N concept detection models are trained such that for each video shot, the n^{th} model outputs a score measuring the likelihood that concept n exists in that video shot. For this part, all kinds of classifiers can be employed to generate different prediction scores, which may

lead to different score-based correlations from the same dataset. That is, for the same C_T (like "cow") in the TRECVID dataset, different hierarchies can be generated based on different classifiers applied. In the aforementioned example, ten score vectors of the positively related concepts and ten score vectors of the negatively related concepts are put together to train an integration model.

In the testing step, each testing video shot of the target concept is plugged into all concept detection models to generate the corresponding testing scores for the related concepts chosen. These scores are then input to the trained score integration model to generate a new set of re-ranked scores. Please note that the scores of the target concept may or may not be used, as shown in Figure 4.5, depending on whether they are chosen in the training phase or not. Finally, the new output scores are evaluated.

4.3 Experiments and Results

4.3.1 TRECVID Dataset

In the experiment, the IACC.1.A and IACC.1.B datasets are chosen from the semantic indexing (SIN) task of the TRECVID 2015 benchmark [165], which aims to detect the semantic concept contained within a video shot. The task assign IACC.1.A as the training dataset and IACC.1.B as the testing dataset. There are several challenges for the SIN task, such as data imbalance, scalability, and the semantic gap [166, 167] as mentioned earlier.

The TRECVID conference series encourage research in information retrieval and provide a huge number of videos for training, and there are more than 300 hours in IACC.1.A and IACC.1.B datasets. By extracting keyframes from each video shot,

totally 262,911 training data instances are generated. We further divide the IACC.1.B dataset into a validation set with 68,663 data instances and a testing set with the same number of data instances [168].

In this dataset, totally 346 concepts are given, including many popular semantic concepts like "Face", "Vehicle", and "Violent" which are common and appear in many research papers. The list of concepts and the detailed explanations can be found in [53]. In this framework, we download the detection scores from the DVMM Lab of Columbia University [169] for all video shots, who ranked the first several years in the TRECVID competition. The TRECVID 2015 training labels are also utilized to increase the number of ground truth in the negative association selection component. The proposed multimedia big data mining system is tested using some of the results from previous work in [109, 110].

4.3.2 Experimental Results

Since we target on imbalanced concept retrieval in this framework, 20 most rare concepts with an average P/N ratio of 0.0001 are chosen. Among the video shots in the testing dataset, each of them have no more than 10 video shots in the dataset. These 20 concepts are: "Car Crash", "Cigar Boats", "Crustacean", "High Security Facility", "Helicopter Hovering", "Cetacean", "Military Buildings", "Rpg", "Prisoner", "Police Truck", "Colin Powell", "Earthquake", "Oil Drilling Site", "Rescue Helicopter", "Dolphin", "Security Checkpoint", "Fire Truck", "Whale", "Cows", and "Yasser Arafat".

The experimental results are shown in Table 4.1. The "Baseline" one is calculated using the raw scores directly from the classifiers in [169]. Though the scores here were

Table 4.1: Experimental results on 20 most rare concepts

MAP / Framework	MAP10	MAP20	MAP50	MAP100	MAP200	MAP500
Baseline	0.0446	0.0438	0.0312	0.0318	0.0322	0.0302
SVM (LL)	0.0125	0.0125	0.0125	0.0125	0.0130	0.0130
SVM (SL)	0.0167	0.0167	0.0167	0.0088	0.0088	0.0090
NB (LL)	0.0056	0.0142	0.0142	0.0124	0.0124	0.0124
NB (SL)	0.0056	0.0146	0.0146	0.0113	0.0132	0.0137
RF (LL)	0.0375	0.0408	0.0297	0.0215	0.0215	0.0215
RF (SL)	0.0426	0.0460	0.0401	0.0417	0.0417	0.0417
LR (LL)	0.0234	0.0309	0.0335	0.0278	0.0256	0.0230
LR (SL)	0.0467	0.0532	0.0554	0.0551	0.0535	0.0529
DAC (LL)	0.0532	0.0577	0.0554	0.0485	0.0462	0.0436
DAC (SL)	0.1130	**0.1130**	0.0856	0.0733	0.0711	0.0614
Proposed	**0.1321**	0.1101	**0.0916**	**0.0885**	**0.0850**	**0.0681**

the best prediction scores, it still performs bad on rare concept retrieval because of the extremely skewed distributions. As mentioned in Section 4.2.3, we use different classifiers including Support Vector Machine (SVM), Naive Bayes (NB), Random Forest (RF), Logistic Regression (LR), and Discriminant Analysis Classifier (DAC) to re-rank those scores. "LL" is for label-label correlations, which means only using the correlations calculated by the label information in the training dataset. Comparatively, "SL" stands for score-label correlations, which is the main contribution of this framework.

The results clearly show that if the target concepts are extremely rare, using only correlations calculated by the label information from the training dataset does not help and can even downgrade the results. Table 4.1 shows that we achieve a better score enhancement when using the information generated by the score-label correlations, in comparison with that using the label-label correlations for every classifier. Albeit with the imprecise raw scores on rare concepts, the proposed framework can successfully re-rank and enhance the results as can be seen from Table 4.1.

Since the Naive Bayes (NB) approach is based on applying the Bayes' theorem with strong independence assumptions between the attributes, which is not true in this case (inter-concept correlations), it performs the worst. Furthermore, because of the nature of Random Forrest (RF) (i.e., random tree selected), we run it three times and the results are averaged. The proposed framework is presented in the "Proposed" column which also includes information from negative correlations. It uses the correlations found in Section 4.2.2 and integrates the scores from those negative-related concepts.

4.4 Conclusions

Rare concept retrieval is a challenge task due to the nature of the imbalanced datasets. Since the data instances in the majority class usually overshadows those in the minority class, it is hard to get acceptable retrieval results when the target concept is a rare concept. In this section, we propose a score re-rank system using the label-score correlations to leverage the semantic concept retrieval task from the video shots. The experimental results clearly show the effectiveness of the proposed framework and how it can successfully enhance the prediction scores of the rare concepts.

The label-score correlations also work like inference rules which can provide a clue for how to define a rare concept. Suppose we have the data of an unknown kind of "animal", using the proposed framework can help find the relationships between several known animals and concepts with it. Considering the "cow" example, we can now better answer the question of what is a "cow". Similarly, we can somehow define a new concept, a new species, or a new object, even though we do not know what it is now. This kind of definitions is very helpful to the fields of information retrieval and

knowledge discovery, which can be further investigated as the future work. Another research direction is to find an efficient way to build larger concept hierarchies. When we have thousands of concepts in a dataset, the trees will be much more complicated and thus more research efforts are needed.

CHAPTER 5

Classifier Fusion and Score Integration by Judgers

In this chapter, a novel idea of classifier combination is proposed and organized as follows. In Section 5.1, the proposed classifier fusion model [170] is introduced in details. In the experimental section, two benchmark action datasets are used for evaluation. Finally, concluding remarks are presented in the last section.

5.1 Framework

5.1.1 Classifier Ensemble

Suppose we have an instances x, where x is a d-dimensional feature vector. Let $\omega_1, \omega_2, \cdots, \omega_M$ be M categories, and $\alpha_1, \alpha_2, \cdots, \alpha_M$ be a finite set of possible actions. Suppose we have totally N classifiers, namely $c_1, c_2, \cdots, c_N$. Each classifier will generate a posterior probability $P_{c_n}(\omega_j|x)$ for x. Here, we define a loss function $\lambda(\alpha_i|\omega_j)$ which describes the loss occurred for taking action α_i when the state of nature is ω_j. Obviously, we can get a set of posterior probabilities used for classification generated by different classifiers as: $P_{c_1}(\omega_j|x), P_{c_2}(\omega_j|x), \cdots, P_{c_n}(\omega_j|x)$.

For each probability function, the expected loss associated with taking action α_i is defined in Equation (5.1). Then, as a classification problem, a zero-one loss function is defined in Equation (5.2).

$$R_{c_n}(\alpha_i|x) = \sum_{j=1}^{M} \lambda(\alpha_i|\omega_j) P_{c_n}(\omega_j|x) \tag{5.1}$$

$$\lambda(\alpha_i|\omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}, \; where \; i, j = 1, 2, \cdots, M \tag{5.2}$$

Using these definitions, for each classifier, the condition risk for category ω_j is defined in Equation (5.3). R needs to be minimized to achieve the best performance for a certain classifier c_n using Equation (5.4) as follows.

$$R_{c_n}(\alpha_i|x) = \sum_{j \neq i} P_{c_n}(\omega_j|x) = 1 - P_{c_n}(\omega_i|x) \tag{5.3}$$

$$R_{c_n} = \int_{x \in \Omega} R_{c_n}(\alpha|x) p(x) dx = \int_{x \in \Omega} \left[1 - P_{c_n}(\omega|x)\right] p(x) dx \tag{5.4}$$

Considering N different classifiers, most previous fusion methods use a certain algorithm to fuse different $P_{c_n}(\omega_j|x)$ for M categories. For example, using the weighted combination rules, we can generate a combined posterior and a new conditional risk R using Equations (5.5) and (5.6).

$$P_{fusion}(\omega_j|x) = \sum_{n=1}^{N} w_n P_{C_n}(\omega_j|x) \tag{5.5}$$

$$\begin{aligned} &R_{fusion} = \textstyle\int_{x \in \Omega} \left[1 - P_{fusion}(\omega|x)\right] p(x) dx \\ &= \textstyle\int_{x \in \Omega_1} \left[1 - P_{fusion}(\omega_1|x)\right] p(x) dx + \int_{x \in \Omega_2} \left[1 - P_{fusion}(\omega_2|x)\right] p(x) dx \\ &+ \; \cdots \; + \textstyle\int_{x \in \Omega_M} \left[1 - P_{fusion}(\omega_M|x)\right] p(x) dx, \\ &where \; \Omega_1 \cup \Omega_2 \cup \cdots \cup \Omega_M = \Omega \; and \; \Omega_i \cap \Omega_j = \phi \; (i \neq j, \; i, j = 1, 2, \cdots, M) \end{aligned} \tag{5.6}$$

As discussed in Chapter 2, the issue of earlier ensemble models is that we can only fuse classifiers performing well in all categories while integrating a relatively

bad classifier may lead to even worse results and eventually reduces the performance in most of the time. However, as a bad guy with a good point, a relatively bad classifier may outperform a good classifier for a certain class. Thus, the proposed framework can still use it to enhance the classification result even though it is not a good choice for all the other classes. We did this by splitting a classifier is split into different "judgers", with each judger working independently to determine the label of a testing instance. We can thus find the good point in a bad classifier and the conditional risk can be reduced by using different posteriors for different classes, as shown in Equation (5.7).

$$\begin{aligned} R_{\min} = & \int_{x\in\Omega_1} \left[1 - P_{\max}(\omega_1|x)\right] p(x)dx + \int_{x\in\Omega_2} \left[1 - P_{\max}(\omega_2|x)\right] p(x)dx \\ & + \quad \cdots \quad + \int_{x\in\Omega_M} \left[1 - P_{\max}(\omega_M|x)\right] p(x)dx \end{aligned} \tag{5.7}$$

5.1.2 Generation of Judgers

In order to generate judgers, the proposed classifier fusion model firstly split a dataset into three parts including a training, a validation, and a testing dataset. The classification models are then trained using the training dataset, followed by the calculation of precision and recall on the corresponding validation dataset. Here, precision is defined as a positive judger and recall rate is denoted as a negative judger, where TP stands for the true positive value, and FP and FN are the false positive and false negative values, respectively as shown in Equation (5.8) and (5.9).

$$J_{pos} = precision = \frac{TP}{TP + FP} \tag{5.8}$$

$$J_{neg} = recall = \frac{TP}{TP + FN} \tag{5.9}$$

Suppose there are M classes in a certain dataset. For one type of features f_l ($l \in [1, L]$, L is the number of feature descriptors, which is two in this framework), based

on the classification results on the validation dataset, $2 \times M$ judgers will be generated for a certain classifier c_n ($n \in [1, N]$, N is the total number of the classification models built on one type of features) as follows:

$$\begin{aligned} &J^{n,l}_{pos_1}, J^{n,l}_{pos_2}, J^{n,l}_{pos_3} \cdots J^{n,l}_{pos_m} \cdots, J^{n,l}_{pos_M} \\ &J^{n,l}_{neg_1}, J^{n,l}_{neg_2}, J^{n,l}_{neg_3} \cdots J^{n,l}_{neg_m} \cdots, J^{n,l}_{neg_M} \end{aligned} \tag{5.10}$$

Here, $J^{n,l}_{pos_m}$ is a positive judger generated by classifier c_n using feature f_l for the class ω_m ($m \in [1, M]$, M is the total number of categories). Correspondingly, $J^{n,l}_{neg_m}$ is a negative judger. If $J^{n,l}_{pos_m}$ is high, it indicates that this judger is relatively accurate. Accordingly, if it judges a testing instance as in class ω_m, it is highly likely that this judgment is correct. On the other hand, if $J^{n,l}_{neg_m}$ is high, it can be considered as a good negative judger since if classifier c_n does not label a testing instance as class ω_m, the ground truth of the instance is not likely to be ω_m. These judgers form the committee to give the final classification results. In summary, suppose there are totally N classifiers and L types of features fed for each classification model, the total number of judgers is $2 \times M \times N \times L$. As an instance, for the UCF11 dataset [127] used in this work, there are 11 classes. Considering all the three types of classifiers introduced on two kinds of features, the total number of judgers generated would be 132 equal to $2 \times 11 \times 3 \times 2$.

5.1.3 The Classifiers Fusion Model

After both positive and negative judgers are generated, the next important issue is how to fuse the outputs from different judgers to draw the final conclusion. In order to assemble these judgers, a novel classifier ensemble framework is proposed as shown in Figure 5.1.

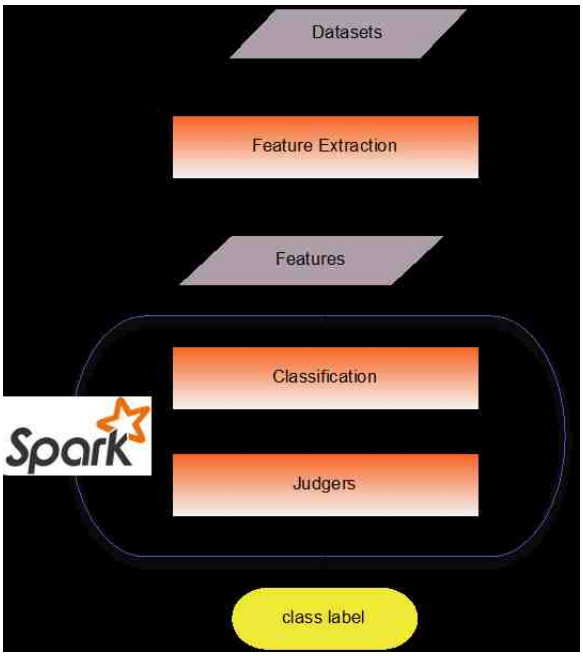


Figure 5.1: The proposed classifier ensemble framework

As shown in Figure 5.2, all the judgers generated are used to build a novel classifier fusion model. For a testing instance x, each classifier will assign x a serial of positive and negative judgers on each feature space. Suppose we have the following list of judgers: $J^{11}_{pos_1}, J^{11}_{neg_1}, J^{11}_{pos_2}, J^{11}_{neg_2}, \cdots, J^{21}_{pos_1}, \cdots J^{31}_{pos_1}, \cdots J^{32}_{pos_M}, J^{32}_{neg_M}$.

Here, the positive judgers are first used and will be re-ranked by their accuracies. When the highest positive judger ranks the first and assigns x as class ω_m, x will be determined as class ω_m. For example, for a testing instance, if $J^{32}_{pos_1}$ is the largest positive judger with the highest accuracy, that testing instance will be classified as class 1.

In the next step, the highest positive judger will be compared with the largest negative judger for the same class, i.e., $J^{n,l}_{neg_1}$ in the previous example. Take the same example, if the value of $J^{21}_{neg_1}$ is larger than $J^{32}_{pos_1}$ (meaning that the negative judger dominates the classification), x won't be assigned to class 1 because there exists a

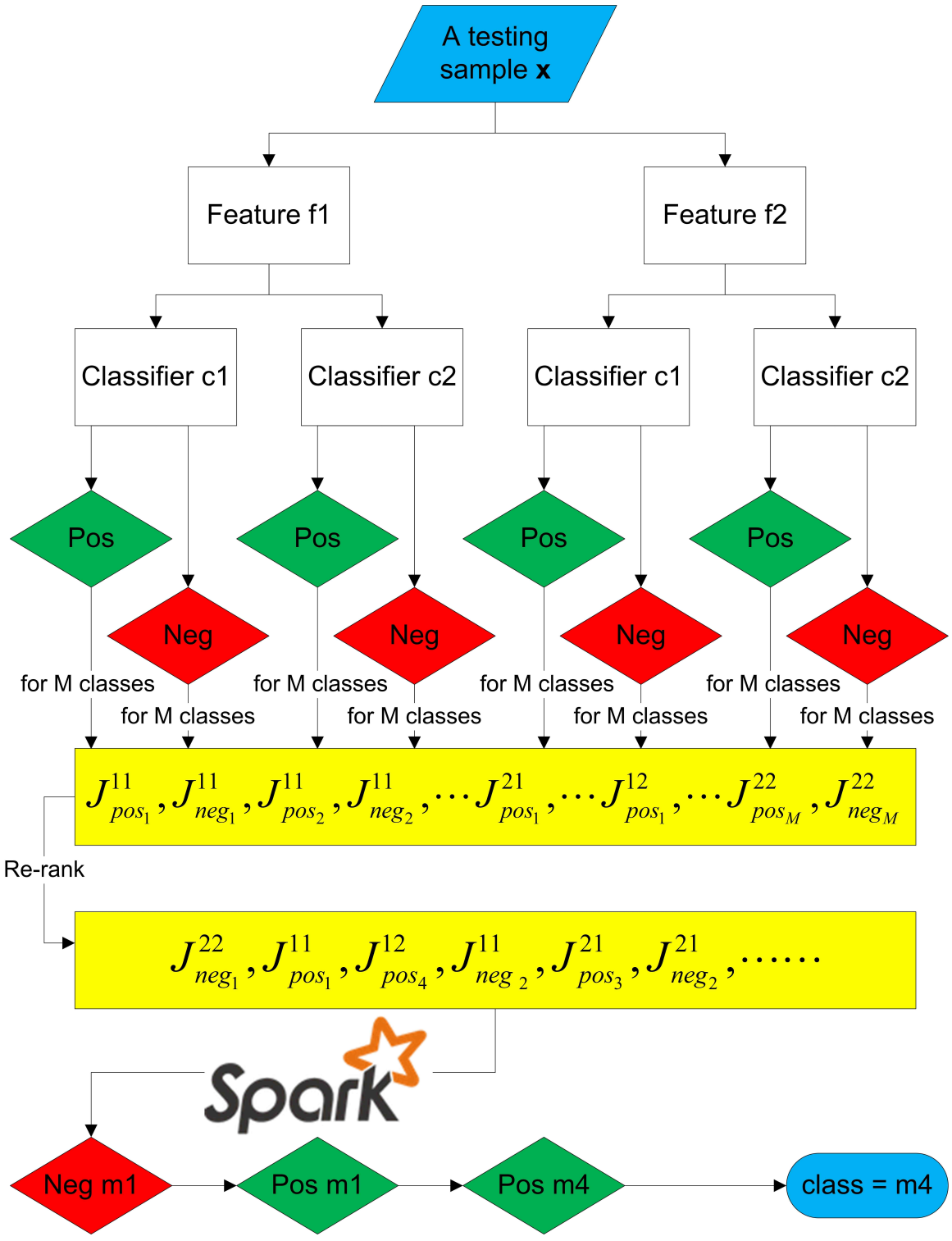


Figure 5.2: An example of the proposed classifier ensemble framework

larger negative judger. That is to say even if the largest positive judger assigns a testing instance to class ω_m, it will be skipped and the second largest positive judger followed will be considered and compared with the corresponding largest negative judger. A similar process continues until a positive judger assigns x to a class without the corresponding largest negative judger rejecting it. The results of this particular testing instance are shown as follows:

$$\begin{aligned} & J^{32}_{pos_1}, J^{12}_{pos_3}, J^{31}_{pos_8}, J^{22}_{pos_2}, \dots \\ & J^{32}_{pos_1} < J^{21}_{neg_1} \\ & J^{12}_{pos_3} > \max J^{n,j}_{neg_3} \end{aligned} \tag{5.11}$$

In this example, the proposed model assigns x to class 3. This procedure is applied to all the testing instances. In very rare cases, however, it should be noticed that if all the corresponding largest negative judgers reject the classification results from the positive judgers, the testing instance will be assigned back to the decision of the highest positive judger at the beginning.

5.2 Proposed Apache Spark Cluster

Nowadays, a number of large-scale data processing frameworks are turning towards generalized MapReduce frameworks after Apache Hadoop is released. Among them, Spark is an open source big data processing framework advertised as extremely fast cluster computing and increases the capability of conventional MapReduce use-cases. It is a fast and general engine for big data processing and has been deployed for many popular large-scale systems. With cores complemented by a set of higher-level libraries including Spark Streaming, SparkSQL for NoSQL database, GraphX for graphs computation, and Mllib for machine learning.

Based on Spark, an efficient system for classifier fusion of large-scale data is built as shown in Figure 3.3. Spark Core is the foundation of the overall project which provides the distributed task dispatching, scheduling, and basic I/O functionalities. Currently, the Spark cluster contains four boxes with one master node and three slave nodes. Each node is setup to instantiate 2 workers with 4 GB of memory and 1 TB of storage.

The main abstraction Spark provided is a Resilient Distributed Dataset (RDD) which is an immutable fault-tolerant, distributed collection of objects that can be operated in parallel. An RDD can contain any type of objects and is created by load-ing an external dataset or distributing a collection from the driver program. In addi-tion to the only two operations in MapReduce, Spark provides many other operations called transformations such as map, sample, groupByKey, reduceByKey, union, join, sort, mapValues, and partionBy. The above operations can be used either stand-alone or in combination to run in a single data pipeline use case. Currently, Spark is originally written in Scala and now fully supports Java; while it also supports Python unstably. In this framework, the codes are all written in Java.

In details, we first read the keys (sample ID) and values (scores from different classifiers for different features) as follows to build a very efficient multimedia large-scale data classification model using Spark:

$$
\begin{aligned}
&Key = sample_1, Value = (s_1^{c_1,f_1,\omega_1}, s_1^{c_1,f_1,\omega_2} \cdots s_1^{c_2,f_1,\omega_1} \cdots s_1^{c_N,f_L,\omega_M}) \\
&Key = sample_2, Value = (s_2^{c_1,f_1,\omega_1}, s_2^{c_1,f_1,\omega_2} \cdots s_2^{c_2,f_1,\omega_1} \cdots s_2^{c_N,f_L,\omega_M}) \\
&Key = sample_3, Value = (s_3^{c_1,f_1,\omega_1}, s_3^{c_1,f_1,\omega_2} \cdots s_3^{c_2,f_1,\omega_1} \cdots s_3^{c_N,f_L,\omega_M}) \\
&\cdots \\
&Key = sample_P, Value = (s_P^{c_1,f_1,\omega_1}, s_P^{c_1,f_1,\omega_2} \cdots s_P^{c_2,f_1,\omega_1} \cdots s_P^{c_N,f_L,\omega_M})
\end{aligned}
\tag{5.12}
$$

Here, the meanings and notations are the same as introduced in previous sections and the output values are the classification results. To easily compare the performance with other existing approaches, two popular medium-sized video datasets are used in the experiments. Nevertheless, the Spark cluster is able to handle large-scale multimedia datasets easily. For a larger dataset, more key-value pairs will be created and thus the system can help more in terms of efficiency in comparison to classical classifier fusion models.

5.3 Experimental Results

To test the efficiency of the proposed classifier ensemble framework, two popular and widely accepted benchmark multimedia datasets in the field of human action recognition are used in the experiments. These two datasets are the KTH dataset [171] and the UCF11 dataset [146].

In the two experiments on KTH and UCF11, the 25-fold cross validation is adopted. Three classifiers introduced in Section 5.3.2 are used, namely SVM, SRC, and HDC; while both SIFT and STIP features are used.

5.3.1 Feature Extraction

Different from some other papers that extract features from the whole images or frames, we do feature extraction only from the Region of Action (ROA) in order to capture the action related information. Here, we use the ROA selection and feature extraction strategy from [66] which improve the action detection and recognition performance in an automated system by fully exploring the ROAs. This cited work

also analyses and integrates the motion information of actions in both temporal and spatial domains.

This approach can be roughly divided into three steps. In the first step, ROAs are driven from two popular spatio-temporal methods including Harris3D corners and optical flow. Next, the idea of integral image in [172] is utilized for its fast implementation of the box type convolution filters. Similar idea is also used in SURF [173] promoted from SIFT [148]. Finally, the Gaussian Mixture Models (GMM) are applied sequentially in this framework. All the mean vectors of the Gaussian components in the generated GMM model are concatenated to create GMM supervectors for video action recognition. SIFT and STIP [174] features which are widely used are extracted from frames in video datasets in this work to describe the action sequences in video action recognition. Other good features can be also fed to the proposed model for better results.

5.3.2 Classification

Similar as features, the classifiers fusion framework accepts most kinds of classifiers. Specifically, three popular classification algorithms are used in this framework as follows:

Support Vector Machine

SVM (Support Vector Machine) is one of the state-of-the-art algorithms for classification in the data mining area. The general idea is to build a separating hyperplane to classify the data instances so that the geometric margin is maximized. In order to handle the case that the classes are linearly inseparable. The kernel trick is utilized.

In this framework, we applied the LibSVM, which is one of the most popular off-the-shelf software implementations. The radial basis function (RBF) kernel is chosen based on experimental results of empirical study.

Sparse Representation

Sparse representation [175] is a hot research topic in the past decade which builds overcomplete dictionaries to represent the training dataset. With this kind of dictionaries including prototype signal-atoms, signals can be described by sparse linear combinations of these atoms. Sparse representation has been widely used in the areas of image denoising, object detection, semantic concept retrieval, information compression and other useful applications including multimedia data classification.

Here, we use Sparse Representation Classification (SRC) [176–181] along with its dictionary learning techniques and design a framework to analyse actions of one person and events between multiple people. For the task of dictionary learning, the widely-used K-SVD algorithm is adopted, which aims at deriving the dictionary of sparse representation using the Singular Value Decomposition (SVD). The class label of a testing instance can be determined by finding the minimum reconstruction error of the testing sample represented by the trained dictionaries. A disadvantage of the SRC scheme is its high computational complexity associated with the minimization problem.

Hamming Distance

A novel scheme Hamming Distance Classification (HDC) [182,183] is also included in this work. HDC is an efficient classifier for real-time applications due to its effi-

ciency. For each class, a threshold will be calculated by the median value of the inner products of each pair of features in the training set. Given a testing instance, a binary string can be coded based on the inner product between the testing sample and each training sample. If the inner product of the testing sample and a training sample is less than the trained threshold, it would be assigned to "0"; otherwise, it would be coded as "1". Finally, the hamming distance between the bit vector from the testing instance and each class is calculated, and this testing sample is assigned to the class with smallest hamming distance.

5.3.3 Results on the KTH Dataset

The KTH dataset includes 6 different human actions (i.e., boxing, hand clapping, waving, jogging, running, and walking) from 25 actors in 4 kinds of scenarios (namely indoors, outdoors, outdoors with scale variation, and outdoors with different clothes). Thus, there are 600 video sequences in total. All videos are in the "avi" format and were recorded in a controlled setting with slight camera motion and a simple background as shown in Figure 5.3.

Table 5.1 shows the confusion matrix of experimental results. To fully evaluate the model, the proposed fusion strategy is compared with 6 other fusion strategies in terms of accuracy and the results are given in Table 5.2. The arithmetic mean and geometric mean represent the sum and product of the scores. Two ways of hybrid means are also tested. One is first to calculate the arithmetic mean scores among different classifiers based on the same kind of features and then to compute the geometric mean scores between different kinds of features; the other kind is to do the opposite.

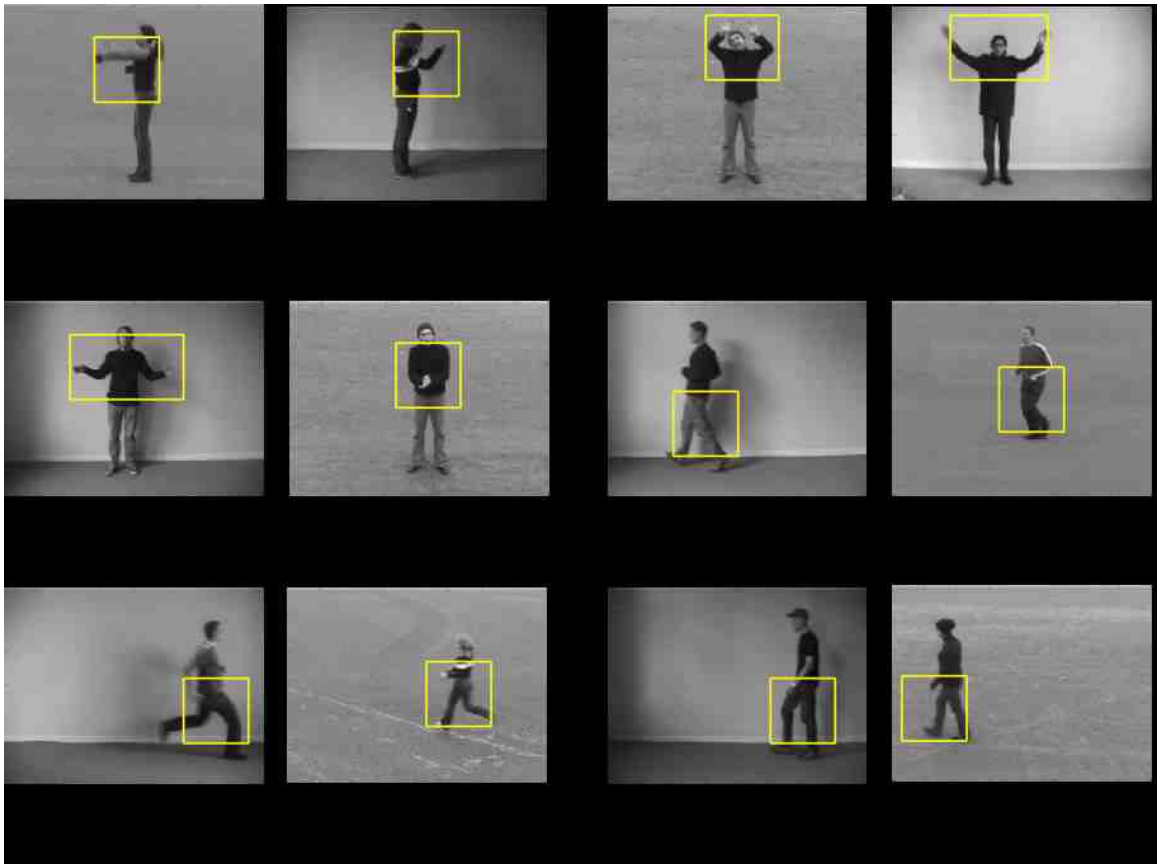


Figure 5.3: Sample frames from the KTH dataset

Table 5.1: The confusion matrix of six action categories in the KTH dataset

MAP / Framework	Box	Clap	Wave	Jog	Run	Walk
Box	**96**	3	0	0	0	1
Clap	0	**99**	0	0	0	1
Wave	1	5	**94**	0	0	0
Jog	0	0	0	**88**	11	1
Run	0	0	0	0	**100**	0
Walk	0	0	0	0	0	**100**

Table 5.2: Comparison of the proposed classifier ensemble framework and other fusion algorithms on the KTH dataset

Algorithm	Average Precision
Arithmetic mean	90.7%
Geometric mean	90.0%
Hybrid mean (Arithmetic for different features)	90.7%
Hybrid mean (Geometric for different features)	90.3%
Linear regression on SIFT	90.5%
Linear regression on STIP	91.2%
The proposed strategy	**96.2**%

In addition, the proposed classifier ensemble model is compared to 4 other published frameworks. Table 5.3 compares the accuracies among them. We also split the experiment by first using only SIFT and then using only STIP as described in Section 5.1. As can be clearly seen from the comparison results, the proposed model performs better the other ones.

Table 5.3: Comparison of overall average precision of the proposed method and state-of-the-art methods on the KTH dataset

Group	Average Precision
Schuldt et al. [171]	71.5%
Dollar et al. [184]	80.7%
Yin et al. [185]	82.0%
Niebles et al. [186]	91.3%
The proposed work on SIFT	93.7%
The proposed work on STIP	95.3%
The proposed work on both features	**96.2**%

5.3.4 Results on the UCF11 Dataset

The UCF11 dataset which also known as "YouTube Action Dataset" is more challenging than the KTH dataset, since it contains realistic actions, camera motions, and complicated backgrounds. There are eleven action categories, namely basketball shooting, biking/cycling, diving, golf swinging, horseback riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog. For each category, the videos are grouped into 25 groups with more than 4 action clips in it. Some sample frames are given in Figure 3.5.

Similar to the experimental steps for the KTH dataset, we compare the proposed work with 6 other fusion strategies in terms of accuracy and the results are given in Table 5.4. Meanwhile, we also compare the results with 4 other state-of-the-art models and the results are shown in Table 5.5. Table 5.5 also shows the advantage

of the proposed model which can fuse different kinds of features to achieve a better performance than the other methods, though the result by only one kind of feature may not able to always outperform other methods.

Table 5.4: Comparison of the proposed ensemble model and other fusion algorithms for the UCF11 dataset

Algorithm	Average precision
Arithmetic mean	74.91%
Geometric mean	74.82%
Hybrid mean (Arithmetic for different features)	75.27%
Hybrid mean (Geometric for different features)	75.09%
Linear regression on SIFT	77.82%
Linear regression on STIP	75.91%
The proposed strategy	**80.3**%

Table 5.5: Comparison of overall average precision of the proposed method and state-of-the-art methods for the UCF11 dataset

Method	Average precision
Chen et al. [187]	67.5%
Perez et al. [123]	68.9%
Liu et al. [147]	71.2%
Mota et al. [188]	75.4%
Multi-classifier on SIFT	69.1%
Multi-classifier on STIP	74.8%
The proposed work on both features	**80.3**%

5.4 Conclusions

Recently, ensemble learning models have received a lot of attentions with the at-tempt to take advantages of multiple useful classifiers. In this section, a novel classifier ensemble framework based on judgers is proposed to fuse the classification results generated from different classifiers and features. The proposed framework is built on an Apache Spark cluster and applied to two different human action video datasets as a proof-of-concept. The experimental results show that the proposed framework outperforms several existing state-of-the-art classification approaches.

Although these two datasets are medium-sized, the proposed framework is capable of handling big datasets as it is built on Spark. Thus, the proposed framework can be easily extended to work with more classifiers and features for other multi-class and multi-feature classification problems. Theoretically, the more classifiers and features included, the more judgers will be generated correspondingly, which can potentially lead to better classification results.

CHAPTER 6

Applications

Several frameworks have been proposed using the deep learning techniques that show promising results in application domains such as automatic speech recognition [189], computer vision [190], and Natural Language Processing (NLP) [191]. This chapter is organized as follows. Utilizing the proposed framework in text data analysis, Section 6.1 introduces its application on an Amazon review dataset while Section 6.2 shows a novel fast stance analysis model for the 2016 United States presidential election campaign details. In Section 6.3, an online information retrieval system is presented for demonstration purpose.

6.1 Imbalanced Text Data Classification

In this section, we will investigate how the proposed deep learning framework performs when dealing with the imbalanced text data with the help of NLP.

6.1.1 Motivation

In the previous sections of this dissertation, we mainly focus on image and video data while the data imbalance problem also exists in text data. For example, the

analysis of reviews from online reviewers is one of the most important research topics. Reviews are the prime mover of goods on e-commerce websites as they impact nearly 20% of the online sales. With global e-commerce product sales pegged at $2 trillion, i.e., 8.3% of US retail sales of the fourth quarter, consumer reviews translate to nearly $400 billion according to the Department of Commerce [192]. Interestingly, these voices sprout from only 5% to 10% of online customers who actually write reviews.

Meanwhile, useful reviews often fall behind due to the mixed sentiments towards the product. Given a 4-star rating, buyers often overlook the details concerning the service, quality, pricing, and other notable services tagged with the product. By applying machine learning techniques on the reviews, buyers would know the reason why the product lost 1 star and the merchant could improve their parts/service/quality correspondingly. Therefore, we propose to develop a novel framework that learns the reviews from multiple categories and breaks down each review to establish baseline averages of the following types:

1. Pricing
2. Quality
3. Delivery and Packaging

6.1.2 Challenges

Text classification is a classical and important research topic in NLP and machine learning with many applications. Currently, the most advanced and efficient way to handle this challenging situation would be to employ solutions from other domains of machine learning such as deep learning. While some previous approaches on deep

learning were applied to the Amazon review dataset [193] and relatively good performance was reported in comparison to the traditional Bag-of-Words (BoW) and Bag-of-ngrams (n-grams) techniques, they all ignored a vital issue in the dataset. The Amazon review dataset is a topically imbalanced dataset with a skew distribution, i.e., most ratings are 4 or 5 stars, while only a few products get 1 or 2 stars. However, we believe that the reviews for bad products with low ratings are more important, since they usually point out the problems and can help Amazon to eliminate dishonest sellers. Such skewness as in the Amazon review dataset poses a significant challenge in major research problems pertaining to data mining and machine learning as discussed in Section 2.2.

Recurrent Neural Network (RNN) has effective performance on processing sequential data, including text data. Therefore, RNN is also an important approach to solve the text classification problem. Gated RNN is proposed and applied to sentiment classification for texts [194]. The gated RNN takes the sentence vectors generated by either a CNN or Long-Short Term Memory (LSTM) and combines them to form a document-level vector, which is used for classification. This hierarchical structure of gated RNN is also applied to the Hierarchical attention network (HAN) [195], where the attention mechanism is incorporated to implicitly encode the structural knowledge in the document representation and thus to improve the classification performance. Their paper reports that they got 63.6% accuracy on the Amazon review dataset using Recursive Neural Networks (RNN); while Joulin et al. [196] got 60.2% accuracy using fastText. We applied their algorithms to a subset of the Amazon review dataset and got similar results: 55.80% using RNN and 58.47% using fastText. However, 68.69% of the ratings are 5 stars, which means we could easily reach 68.69%

accuracy by assigning 5 stars to all the testing samples though the performance would be meaningless at all. As shown in Tables 6.1 and 6.2, most reviews are 5 stars while many popular deep learning algorithms can hardly retrieve true-positive reviews with 1 star or 2 stars, i.e., most of them are biased towards the majority classes (e.g., 5 stars). When identifying the minor classes, these classifiers often perform inaccurately, even for very large datasets with considerable numbers of training instances (2 million training samples in both cited papers).

Table 6.1: Confusion Matrix of the Amazon review dataset using RNN

Ground truth \ Predicted label	1 star	2 stars	3 stars	4 stars	5 stars
1 star	**0**%	0%	15.52%	6.90%	77.59%
2 stars	0%	**0**%	10.94%	4.69%	84.38%
3 stars	0%	0%	**16.39**%	3.83%	79.78%
4 stars	0.22%	0%	17.51%	**8.53**%	73.74%
5 stars	0.20%	0.07%	12.02%	8.14%	**79.57**%

Table 6.2: Confusion Matrix of the Amazon review dataset using fastText

Ground truth \ Predicted label	1 star	2 stars	3 stars	4 stars	5 stars
1 star	**0**%	0%	3.45%	24.14%	72.41%
2 stars	0%	**0**%	0	17.50%	82.50%
3 stars	0%	0%	**2.61**%	19.61%	77.78%
4 stars	0.21%	0.21%	2.26%	**18.31**%	79.01%
5 stars	0%	0.19%	2.70%	17.97%	**79.14**%

6.1.3 Outline

The rating of the Amazon reviews can be regarded as an application of the text classification problem, where each level of the rating is correspondingly mapped to one class. To simplify this task and make it more suitable for our framework, we polarize the training dataset and make it to a two-class problem, i.e., map "1 star" to positive and "5 stars" to negative. Deep learning methods including RNN, TextCNN, and

their combinations are commonly used to conduct the analysis and classification on text data and have shown a lot of promising results in different applications [194–197]. In this application, TextCNN [197] is applied to test our framework in Chapter 2.

After polarization, positive and negative sentences are loaded from the raw data files. Then, the next step in cleaning up text data (e.g., removing special characters) is to have a better dataset for training. The CNN model we use in this dissertation looks roughly as shown in Figure 6.1. The first layer embeds words into low-dimensional vectors. The next layer performs convolutions over the embedded word vectors using multiple filter sizes. In this example, it slides over 3, 4, or 5 words at a time. Finally, we max-pool the result of the convolutional layer into a fixed-length feature vector, add dropout regularization, as well as classify the result using a softmax layer.

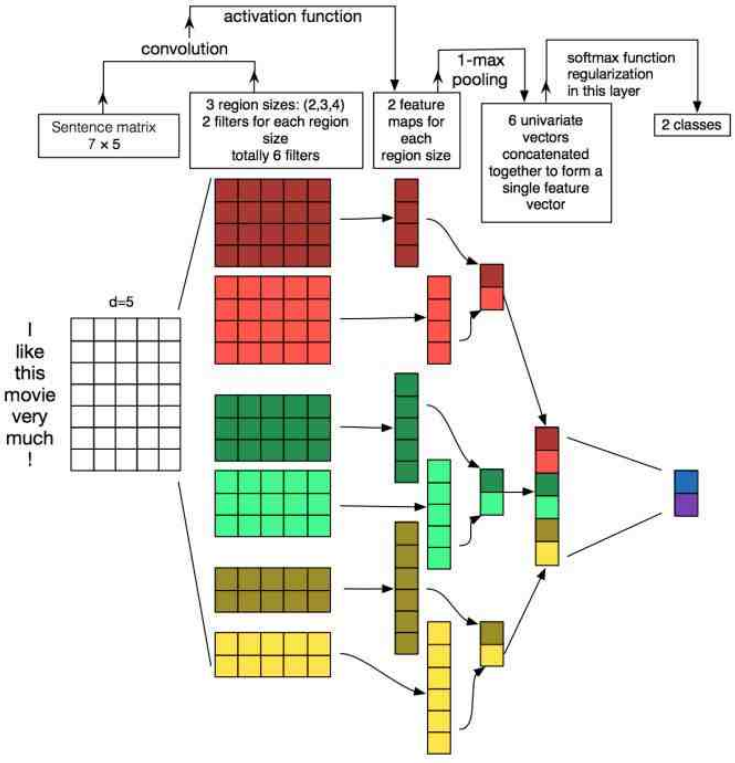


Figure 6.1: Model architecture with one channel for an example sentence

The first layer in our model is the embedding layer, which maps vocabulary word indices into low-dimensional vector representations, i.e., word2vec vectors. It's essentially a lookup table that we learn from the data. Using filters of different sizes, convolutional layers are built followed by max-pooling. To overcome the class imbalanced problem as mentioned in Chapter 2, we plan to build a balanced training set and feed similar numbers of training samples for each class to a CNN model. Basically, more positive training instances (in this case for the "1 star" reviews) would be generated based on their statistical distribution and the same number of training samples from the negative class (i.e., "5 stars" reviews) would be packed and fed to the CNN model. The details are shown in Algorithm 1. Since convolutional layers produce various sizes of vectors by different shapes of filters, we use a 1-maximum pooling layer to merge the results into fixed-length feature vectors. Similar to CNNs in image classification, the dropout layers are applied for regularization, i.e., stochastically "disable" a fraction of their neurons which prevent neurons from co-adapting and force them to learn individually useful features. Especially for an imbalanced dataset, a high dropout rate is necessary to prevent overfitting on the negative samples. Using feature vectors after the max pooling layer with dropout applied, predictions are finally generated by doing a matrix multiplication and picking the class with a higher score. We could also apply a softmax function to convert raw scores into normalized probabilities, which wouldn't change the final predictions.

To learn reviews in terms of multiple categories, one of the most popular algorithms for topic modelling is Latent Dirichlet Allocation (LDA). In the NLP domain, LDA is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. The first

step of LDA is grouping words into topics. Then, a document can be classified to a certain topic based on which group of words it talks about. However, some of the topics retrieved by LDA may not make sense as shown in Table 6.3a. Moreover, only the number of topics can be controlled by LDA rather than the types we designed in Section 6.1.1. In supervised learning, we can go back and check the potential problem in the decision making process while it is not applicable in unsupervised learning. In order to tell the LDA model to split documents into n topics, we can set n groups of "seed" words and let the model converge around those terms [198], which named GuidedLDA. During the initialization step in GuidedLDA, we can give an extra boost to seed words and thus bias them more towards the seeded topics. Table 6.3b lists an example of seed words for seed topics in Section 6.1.1 and the final results, i.e., words selected by GuidedLDA, are shown in Table 6.5a.

Table 6.3: LDA versus GuidedLDA

(a) Top 20 words in each topic selected by LDA

Topic 1	Topic 2	Topic 3
pedal	guitar	strings
sound	one	guitar
amp	well	sound
can	use	like
use	just	great
great	strap	picks
one	stand	good
like	good	just
good	great	string
just	can	pick

(b) Seeded 10-word set in selected topics for GuidedLDA

Pricing	Quality	Delivery
price	quality	delivery
cheap	excellent	early
expensive	reliability	fast
bargain	usability	time
reasonable	durability	package
worth	compatibility	condition
cost	ability	sealed
affordable	poor	whole
\	professional	loose
\	utility	\

Nevertheless, two major issues still remain in GuidedLDA. One is that the topics selected may not be separable, and the other is a document may belong to multiple documents. Different from image and video data which work with high-dimensional datasets encoded as vectors of the individual raw pixel-intensities, NLP systems tra-

ditionally treat words as discrete atomic symbols. However, these encodings are kind of arbitrary, and provide useless information to the system regarding the relationships that may exist between the individual words. Furthermore, representing words as discrete "ids" often leads to data sparsity, which means more data are required to successfully build the statistical models. While Hinton proposed the idea of distributed representations of words in 1986 [199], which states that words appeared in the same contexts share semantic meaning; two papers published in 2013 by Mikolov introduced word2vec [200,201], which is an efficient predictive model for learning word embeddings from raw texts, become popular in the field of NLP. Word2vec comes in two flavors, i.e., the Continuous Bag-of-Words model (CBOW) which predicts target words from source context words versus the Skip-Gram model which predicts source context words from the target words. In this application, we use the seed words in GuidedLDA and apply word2vec to find similar words in raw text in order to find reviews on the three selected topics in Section 6.1.1.

6.1.4 Experimental Results on Amazon Product Data

The dataset used for evaluation contains product reviews and metadata from Amazon, including more than one hundred million reviews from 1996 and has been widely used in a lot of publications [195, 196]. The reviews in this dataset include ratings, texts, and helpfulness votes. In this experiment, reviews from a subset, Musical Instruments, are polarized and used for imbalanced text data classification. To highlight the contribution of this dissertation and exclude irrelevant variables, we use static word vectors and learn our own word embeddings from scratch rather than using pre-trained word2vec vectors. 128 filters are used for each size of the filter, and

the dropout rate is set to 0.9 to maximally prevent the model from being biased the negative class. The comparison of results in terms of AP are shown in Table 6.4 and prove the effectiveness of the proposed framework as it outperforms the regular CNN for most kinds of settings. The second column of Table 6.4 is for the number of filters.

Table 6.4: Comparison of the proposed framework and original TextCNN on the Amazon review dataset

Framework	#	Size	AP10	AP20	AP50	AP100	AP200	AP500	AP1000
Original	4	3-6	0.0000	0.0000	0.0000	0.0000	0.0071	0.0110	0.0143
Proposed	4	3-6	**0.6250**	**0.4722**	**0.2784**	**0.2564**	**0.1986**	**0.1511**	**0.1181**
Original	5	3-7	0.0000	0.0000	0.0000	0.0000	0.0111	0.0151	0.0200
Proposed	5	3-7	**0.7222**	**0.5281**	**0.4556**	**0.3755**	**0.2792**	**0.2144**	**0.1736**
Original	6	3-8	0.0000	0.0000	0.0000	0.0172	0.0172	0.0126	0.0145
Proposed	6	3-8	**0.4958**	**0.4832**	**0.4025**	**0.3334**	**0.2589**	**0.1784**	**0.1452**
Original	7	3-9	0.0000	0.0000	0.0000	0.0000	0.0084	0.0092	0.0113
Proposed	7	3-9	**0.7111**	**0.6695**	**0.5589**	**0.3659**	**0.2731**	**0.2073**	**0.1633**
Original	7	5-11	1.0000	**1.0000**	**0.2746**	**0.2255**	**0.1702**	0.1038	0.0810
Proposed	7	5-11	**1.0000**	0.5588	0.2644	0.2124	0.1497	**0.1143**	**0.0966**
Original	7	7-13	0.6000	0.4237	0.3594	0.2252	0.1721	0.1371	0.1049
Proposed	7	7-13	**0.7667**	**0.6464**	**0.4792**	**0.3573**	**0.2350**	**0.1557**	**0.1198**

In the second stage, a word2vec model is built using the same dataset. Particularly, a skip gram neural network model is trained to map words into vectors. Given the same set of seed words, we find 10 most similar words for each seed word in Table 6.3b based on its word2vec value and the result is shown in Table 6.5b. Each testing instance (document) is labelled as one or the mix of three interested topics, i.e., pricing, quality, as well as delivery and packaging. Since it is an unsupervised learning task and we don't have the ground truth, the same number of testing instances are randomly picked from each topic based on the results by GuidedLDA and the proposed framework. The comparison of the accuracy is depicted in Table 6.6, which proves the effectiveness of our model.

Table 6.5: GuidedLDA versus Proposed

(a) Words selected by GuidedLDA using seed words in Table 6.3b

Pricing	Quality	Delivery
price	quality	time
quality	sound	guitar
good	time	strings
great	pedal	one
money	price	use
well	studio	just
worth	one	like
guitar	can	will
cheap	use	cheap
sound	amp	can
one	good	quality
better	great	good
value	like	get
product	pedals	long
much	just	well
strings	cheap	string
cost	mic	strap
just	get	price
time	will	sound
expensive	guitar	much

(b) Words selected by word2vec in each topic for clustering

Pricing	Quality	Delivery
price	quality	delivery
pricei	qualityi	deliver
pricey	highquality	delivered
pricy	reality	delivering
pricethe	fidelity	delivers
priced	quantity	promptly
prices	qualities	promised
priceless	reliability	overpriced
overpriced	craftsmanship	thoroughly
pricier	conspros	prompt
pricing	utility	promise
cheap	excellent	early
cheapy	excels	nearly
cheaply	exceeds	clearly
...	...	...
expensive	reliability	fast
expense	durability	faster
inexpensive	usability	fasten
...	...	...
...	...	...

Table 6.6: Comparison of the proposed framework and GuidedLDA on Amazon review clustering

Topic \ Accuracy	GuidedLDA	Proposed
Pricing	72.3%	**89.7**%
Quality	82.7%	**93.0**%
Delivery & Packaging	27.7%	**62.3**%

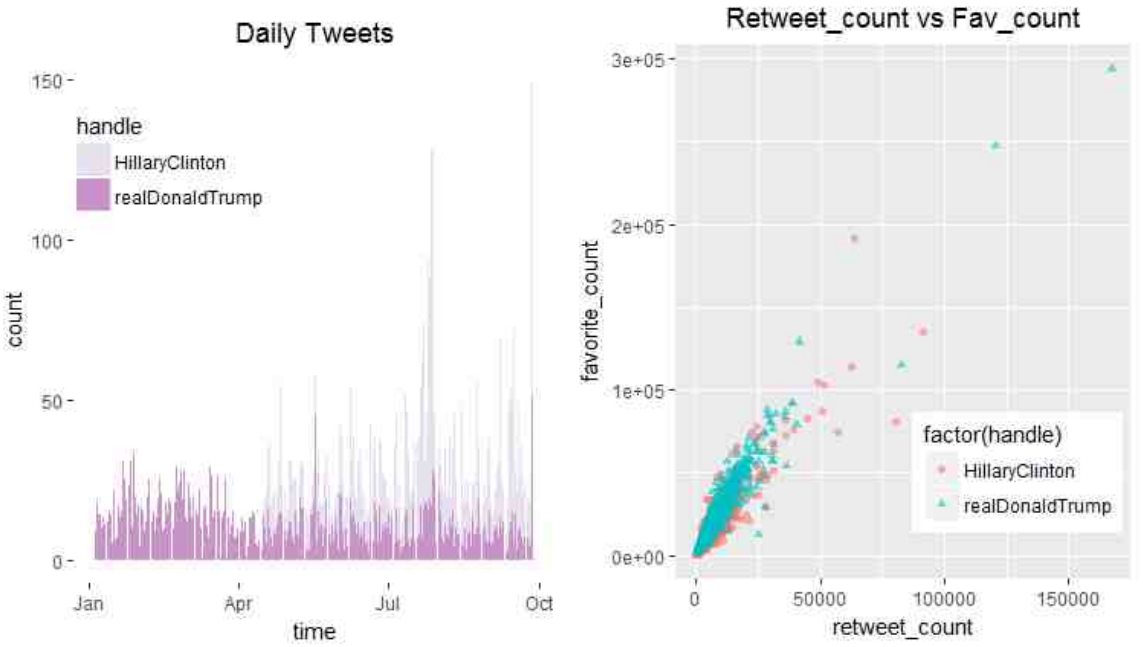
6.2 Efficient Large-scale Stance Analysis in Twitter

This application introduces a novel framework for stance detection in bipolar affinities. The 2016 U.S. presidential election campaign was used as a test use case because of its significant and unique counter-factual properties. The results show that our proposed framework achieves high accuracy when compared to several existing state-of-the-art methods.

6.2.1 Introduction of Stance Classification

In recent years, there has been a major growth in the use of microblogging platforms. Microblogs allow the users to exchange small contents such as short videos, sentences, and links. Some previous research efforts were paid on these kinds of multimedia data [44–46, 64–67, 92]. Twitter is one of the most widely used microblog platforms. Users range from regular users to politicians, celebrities, and company representatives. Therefore, it is possible to collect posts of users from different social and interested groups. Meanwhile, the battle on Twitter is an integral part of a prearranged effort to disturb the 2016 U.S. presidential election. To visualize the overall picture, Figure 6.2a is used to show the daily tweet counts between Hillary Clinton and Donald Trump for the election time period in 2016; while the popularity measure between the two candidates is shown in Figure 6.2b by mapping the retweet and favorite counts of the two candidates.

One important research direction here is stance analysis which implies the political tendency of the public. In this dissertation, "stance classification" is defined as automatically determining whether a Twitter user tends to endorse the candidate of Democratic or Republican Party. By tweets from the Twitter accounts, researchers



(a) Daily tweets of the two candidates

(b) Popularity vs following measure of all supporters grouped by their candidates

Figure 6.2: Daily tweets versus Retweet and favorite counts

can deduce whether a user is either for or against the target. Therefore, another objective of this dissertation is to automatically infer the stances of Twitter users to see whether a user is likely a Hillary Clinton or Donald Trump supporter. While most election predictions reply on polls, automated stance classification can be applied to a much larger number of samples and bring complementary information to predict the election results.

6.2.2 Previous Work on Stance Analysis

Based on our best knowledge, though stance analysis has not been used for election prediction, some earlier work ran experiments that used Twitter hashtags and emoticons such as #bestfeeling, #epicfail, and #news to identify positive, negative, and neutral tweets to train and analyse the sentiment of a tweet [202]. The sen-

timents were identified as a powerful predictor in differentiating the behaviours of various accounts. Agarwal et al. [203] proposed a 3-way task of separating tweets into positive, negative, and neutral, and then used 3 models: unigram, feature-based, and tree kernel-based models to split the data. It was proposed in [204] to use a psychometric instrument to classify six mood states including tension, depression, anger, vigour, fatigue, and confusion. The authors used aggregated Twitter content to compute a six-dimensional mood vector for each day in the timeline. One challenge in Twitter analysis is to identify and collect the right corpus that corresponds well to the domain and context of the tweets. This was attempted in [205] to focus and improve the corpus by an automatic collection and by using TreeTagger for POS-tagging. The wide scale effects of socioeconomic events on the overall general mood of tweets were explored in [204] over the longer periods of time. This provides a useful yardstick to track the sentiments, but this method does not solve the problem of context invariance.

A hypothesis was proposed in [206] that every non-hyperbolic tweet was from Donald Trump's staff while every hyperbolic tweet was from Donald Trump himself. The researchers collected Donald Trump's tweets from Donald Trump's account including the "source" information and found out that most tweets are from either iPhones or Android phones. Their analysis showed that the iPhone and Android tweets are clearly from different people since tweets from them used different hashtags, retweeted in distinct ways, and were posted during different times. They also found that the iPhone tweets were less angry and more positive with benign announcements, while the Android tweets tended to be more negative with angry words. In [207], machine learning techniques were utilized to do sentiment analysis on candidates' Twitter

mentions. They collected millions of tweets posted by users who discussed U.S. politics for Americans and non-Americans worldwide, and classified them based on their sentiment. Each posted tweet related to Hillary Clinton or Donald Trump was labelled with either positive, neutral, or negative. The authors concluded that there were much more negative tweets about both candidates than positive tweets, while there were less tweets that mentioned Hillary Clinton than Donald Trump. In [208], two groups of hashtags were defined arbitrarily, where each group was assumed to support Hillary Clinton or Donald Trump, respectively. After that, the author used descriptive statistics methods and concluded that Donald Trump's campaign knew more about how to use Twitter chat bots than the Hillary Clinton's side.

6.2.3 Twitter Dataset

The rise in popularity of social interacting websites such as Facebook, Twitter, and Snapchat has been challenged by the upsurge of unwelcomed and troubling bodies on these systems. This includes spam senders, malware systems, and other content contaminators. Before we start to introduce our framework, it is noted that highly automated accounts with 450 tweets per day produced almost 18% of entire Twitter circulation in the 2016 U.S. presidential election. Since it is also observed that those disruptive systems called bots are inclined more towards circulating negative news than positive information, we apply a novel framework named Associative Affinity Factor Analysis (AAFA) [209, 210] designed for bot identification which can identify real people from bots in order to remove fake accounts before stance analysis.

In order to do stance analysis for the 2016 U.S. election test use case, a dataset that includes the supporters of both sides is necessary. However, due to privacy is-

sues, it is nearly impossible to get the account names of the supporters. Luckily, Wikipedia provides the lists of Hillary Clinton and Donald Trump presidential campaign endorsements [211]. These lists include "big names" who have publicly claimed their endorsements for the office of the president of Hillary Clinton and Donald Trump as presidential nominees. Since these supporters are notable individuals, the information was reliable and did not change much in the campaign. After data cleaning, 310 supporters of Hillary Clinton and 412 supporters of Donald Trump were included to build the experimental dataset.

In addition, the Twitter API was used to collect 3240 tweets from each supporter with time, resource, retweet, etc. After the data collection, we extracted the details of the supporters' accounts, cleaned the text data from all tweets, and mapped the truncated words to get the hashtag information.

6.2.4 Affinity-based Stance Detection

Consider each hashtag in a tweet as a concept and find the recurring itemset in Donald Trump's retweets or comment feed. If we are able to find multiple instances of people continuously together based on the Association Affinity Network (AAN) [49, 110, 212, 213], then they are bots. The confidence score is replaced by the average sentiment score. It was observed that bots usually have consistently positive or negative sentiments in their tweets. In addition, for real human supporters, individuals who endorsed Hillary Clinton tended to use different hashtags comparing to those supported Donald Trump, and vice versa.

One attempt in the literature [208] built two groups of arbitrary hashtags, from their domain knowledge, to find the Hillary Clinton and Donald Trump supporters.

However, this approach lacks reproducibility and domain invariance. To overcome this challenge and find distinct hashtags, we apply the log odds ratio approach [214]. For a hashtag n, we calculate C_n^H and C_n^T which represent the numbers of times n was used by the Hillary Clinton supporters and Donald Trump supporters. Similarly, U_n^H and U_n^T represent the numbers of distinct accounts of the Hillary Clinton and Donald Trump supporters that used hashtag n. Next, the scores S_n^C and S_n^U are calculated to measure the likelihood values of a hashtag being associated with either of the candidates as shown in Equations (6.1) and (6.2).

$$S_n^C = \log_2 \frac{\frac{C_n^H+1}{\sum_{i=1}^{N} C_i^H+1}}{\frac{C_n^T+1}{\sum_{i=1}^{N} C_i^T+1}}; \tag{6.1}$$

$$S_n^U = \log_2 \frac{\frac{U_n^H+1}{\sum_{i=1}^{N} U_i^H+1}}{\frac{U_n^T+1}{\sum_{i=1}^{N} U_i^T+1}}. \tag{6.2}$$

Here, N refers to the total number of supporters. The scores and the ranked hashtags are given in Table 6.7. For comparison, the hashtag lists are shown in Table 6.8 by the domain knowledge [208] and the tweets from the candidates (i.e., Hillary Clinton and Donald Trump) [207]. It is clear that some unique tashtags can only be automatically found using the proposed framework, e.g., CIR (Comprehensive Immigration Reform Act), RenewUi (federal unemployment extension), RestoreTheVRA (Voting Rights Act), Dobbs (Lou Dobbs), PJNET (Patriot Journalist Network), and VAWA (Violence Against Women Act).

Table 6.7: Ranked hashtags based on the proposed framework (case insensitive)

Rank	# of hashtags used		# of distinct accounts that use a hashtag	
	Hillary Clinton	Donald Trump	Hillary Clinton	Donald Trump
1	*CIR*	*Dobbs*	*RaiseTheWage*	*TrumpPence*16
2	*RenewUi*	*TrumpPence*16	*HoldTheFloor*	*CrookedHillary*
3	*RaiseTheWage*	*PJNET*	*RestoreTheVRA*	*WakeUpAmerica*
4	*ActOnClimate*	*WakeUpAmerica*	*VAWA*	*PJNET*
5	*WomenSucceed*	*TrumpTrain*	*MarriageEquality*	*VoteTrump*
6	*DoYourJob*	*AmericaFirst*	*WorldAidsDay*	*Jesus*
7	*RestoreTheVRA*	*ProLife*	*GunViolence*	*TrumpRally*
8	*DisarmHate*	*TeaParty*	*ProtectOurCare*	*Hannity*
9	*TimeIsNow*	*MakeAmericaGreatAgain*	*StopGunViolence*	*Trump*45
10	*GetCovered*	*ConfirmGorsuch*	*LoveIsLove*	*TrumpPence*2016

Table 6.8: Ranked hashtags based on domain knowledge and tweets from the candidates (case insensitive)

Rank	Domain knowledge		Candidate tweets	
	Hillary Clinton	Donald Trump	Hillary Clinton	Donald Trump
1	*votehillary*2016	*MAGA*	*DemsInPhilly*	*Trump*
2	*VoteHillary*	*HillarysBigotry*	*RNCinCLE*	*MakeAmericaGreatAgain*
3	*NeverTrump*	*CrookedHillary*	*DebateNight*	*VoteTrump*
4	*IAmWithHer*	*Hillary*4*Prison*	*debatenight*	*AmericaFirst*
5	*WeAreWithHer*	*NeverHillary*	*TBT*	*MAGA*
6	*NoTrump*	*TrumpTrain*	*NBCNewsForum*	*ImWithYou*
7	*TrumpLies*	*VoteTrump*	*DemConvention*	*TrumpTrain*
8	*StopTrump*	*LockHerUp*	*WomanCard*	*TrumpPence*
9	*DumpTrump*	*WakeUpAmerica*	*EstoyConElla*	*FITN*
10	*TrumpUnfit*	*TrumpsArmy*	*LoveTrumpsHate*	*GOPDebate*

6.2.5 Results of Stance Classification

To apply the affinity-based stance detection method, the hashtags of people supporting Hillary Clinton and Donald Trump were extracted. The associative affinity was evaluated for one-itemset and two-itemset hashtags occurring in their tweets. The final ranking of the predictive hashtags was ranked according to an empirically selected threshold. Each hashtag itemset has a dynamic threshold but the hashtags with the highest affinities were selected. Table 6.7 illustrates these case insensitive ranked hashtags for the two candidates. The final hashtag lists are selected based on both the "number of a hashtag being used" and the "number of distinct accounts that use a hashtag". The overlapped hashtags are cleaned and finally a list of 128 hashtags is created to generate the feature vectors for the Hillary Clinton and Donald Trump supporters. Based on the number of a hashtag used, a feature vector is generated and normalized for each account.

Table 6.9: Accuracy and F-score comparisons in stance analysis

Classifier	Accuracy	F-score
SVM	0.8089	0.7128
Random Forest	**0.8492**	**0.8156**
DAC	0.7493	0.6190
Linear Regression	0.7812	0.6853
Logistic Regression	0.7867	0.7061

For comparison, our stance classification model is evaluated against several popular classifiers including Support Vector Machine (SVM) [158], Random Forest [161], Discriminant Analysis Classifier (DAC), Linear Regression, as well as Logistic Regression [134]. As shown in Table 6.9, an average accuracy of 80 percent is obtained without any domain knowledge and polls. Random Forest performs the best for this task due to the nature of our feature vectors (i.e., different weights for the hashtags). We could achieve higher accuracy as well as F-score values by removing some testing

instances with the number of hashtags in Table 6.7 lower than a certain threshold, as many people may refuse to disclose their endorsements in polls. Since the number of accounts (i.e., the size of the dataset) is already small, we would prefer not to remove any data instance. Another reason would be explained at the end of this section.

6.2.6 Deep Learning Based Stance Analysis

By applying a similar framework as discussed in Section 6.1.3, this stance analysis task can also be solved by deep learning. After several common pre-processing steps, a TextCNN model for this balanced Twitter dataset is built for binary classification. With an additional max pooling layer to fuse the results from whatever number of tweets in the final stage, the owner of a Twitter account can be labelled as a Hillary or Trump supporter. The experimental results are provided in Table 6.10 with 90 percent accuracy. Nevertheless, there are more than 1 million tweets in this dataset per candidate, which makes the computational complexity in the training procedure very high, in terms of both time and space. Additionally, many tweets are irrelevant with politics. For instance, most tweets of Katy Perry, the biggest celebrity supporter of Hillary, are regarding music. The training effort paid on those tweets are therefore useless and may potentially hurt the model.

Table 6.10: Confusion matrix for stance analysis by deep learning on raw text data

Ground truth \ Predicted label	Hillary supporter	Trump supporter
Hillary supporter	**100**	3
Trump supporter	21	**116**

Using the results from the previous section, we can pick only those tweets with the list of 128 hashtags to build the TextCNN model and thus save 90 percent training time. For a few number of accounts that don't use any of the selected hashtags, we

simply label them as Trump supporters for convenience. The accuracy on the filtered Twitter data is 88.75% as shown in Table 6.11, which is just 1 percent lower than the results on the raw text data.

Table 6.11: Confusion matrix for stance analysis by deep learning on filtered data

Predicted label / Ground truth	Hillary supporter	Trump supporter
Hillary supporter	**81**	22
Trump supporter	5	**132**

Furthermore, the time complexity of the algorithm in the previous section is clearly less than the model built using deep learning. Recall that the classifier fusion framework in Chapter 5, we find the affinity-based method is a good classifier for Hillary supporters by the confusion matrix on the validation set in Table 6.12. By applying the model in the previous section to a validation set, we get 97.3% accuracy on Hillary supporters, which means if the affinity-based classifier considers a testing Twitter account supports Hillary, the owner is highly possible a real Hillary supporter. Also, if the affinity-based method labels the owner of a Twitter account as a Hillary supporter, it doesn't need to go through the testing stage in the deep learning based model.

Table 6.12: Confusion matrix for affinity-based stance detection on validation set

Predicted label / Ground truth	Hillary supporter	Trump supporter
Hillary supporter	**73**	30
Trump supporter	2	**136**

The final experiments are conducted using the framework in Chapter 5 and the results from affinity-based and deep learning based classifiers can be thus integrated together. The final results which prove the efficiency of the overall framework for this application are shown in Table 6.13. Since the time complexity of the affinity-based

method is much less than the deep learning one, 50 percent testing time could be saved approximately (assuming that Hillary has a 50 percent favorable rating).

Table 6.13: Accuracy comparisons on Twitter dataset

Classifier	Accuracy
Affinity-based (RF)	84.9%
Deep learning based (raw text data)	90.0%
Deep learning based (filtered data)	88.8%
Proposed framework	**90.8**%

To show the effectiveness of the proposed framework, a mathematical proof is given below. Let p be the favorable rating of Hillary which is unknown, $\hat{p}$ be the rating in a sample set, and $\hat{\sigma}$ be the estimated favorability. For a two-candidate election poll, based on the Central Limit Theorem (CLT), if a polling organization samples n adults, the 95% confidence interval is:

$$[\hat{p} - 2 \times \hat{\sigma}, \hat{p} + 2 \times \hat{\sigma}]; \text{ where } \hat{\sigma} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \tag{6.3}$$

Let d be the sampling error, i.e., the radius of the confidence interval. Since $\hat{p} > 0.5$, $\hat{p} \times (1 - \hat{p}) < 0.25$. Therefore, we have:

$$1.96 \times \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \leq 1.96 \times \sqrt{\frac{1}{4n}} \leq d; \text{ so } n \geq \frac{1.96^2}{4d^2} \tag{6.4}$$

If $d = 0.03$ or 3%, $n \geq 1067.11 \approx 1068$. If $d = 0.01$ or 1%, $n \geq 9604$. Based on Law of Large Numbers (LLN), $\lim_{n\to\infty} \hat{p} = p$ while there is a trade-off between the sampling error and the number of samples. Therefore, a polling company often only samples one thousand adults. Even if the company can afford the cost to get ten thousand samples, the time cost for the poll would be too long. Meanwhile, 3% is a big number in many swing states, especially Hillary swamps Trump in popular vote for just 2.1%.

Using the proposed framework, assume the classification accuracy is q and n is big enough, p actually follows a Beta distribution as shown in Equation (6.5) or Equation

(6.6). If $\hat{p} = 50\%$, $q = 90.8\%$, and $n = 1000$, a high kurtosis Probability Density Function (PDF) of the Beta distribution is drawn in Figure 6.3 and the corresponding Cumulative Distribution Function (CDF) is drawn in Figure 6.4a. Figure 6.4b shows the CDF when n equals one million, where p has a 99.9999% probability falling into the interval of $50\% \pm 0.3\%$. The model designed in this Section is able to process hundreds of testing samples per minute and thus handling one million samples becomes possible. All in all, the proposed framework in this application could save 90 percent of the training time and 50 percent of the testing time, and generate better classification results.

$$B_{pdf} = \frac{[q \times p + (1-q) \times (1-p)]^{\hat{p} \times n} [(1-q) \times p + q \times (1-p)]^{(1-\hat{p}) \times n}}{\int_0^1 [q \times \dot{p} + (1-q) \times (1-\dot{p})]^{\hat{p} \times n} [(1-q) \times \dot{p} + q \times (1-\dot{p})]^{(1-\hat{p}) \times n} d\dot{p}} \tag{6.5}$$

$$Beta_{pdf}(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 \dot{x}^{\alpha-1}(1-\dot{x})^{\beta-1} d\dot{x}}; \text{ where}$$
$$x = 1 - q - \bar{p} + 2q\bar{p},\ \alpha = np + 1,\ \beta = n - np + 1 \tag{6.6}$$

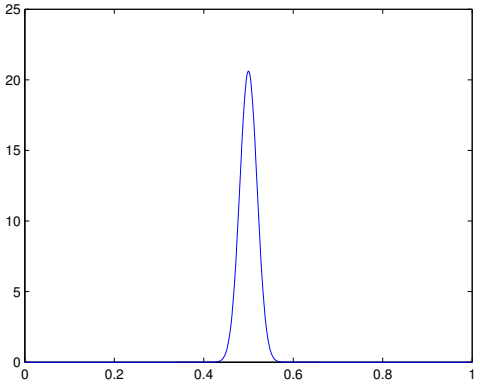
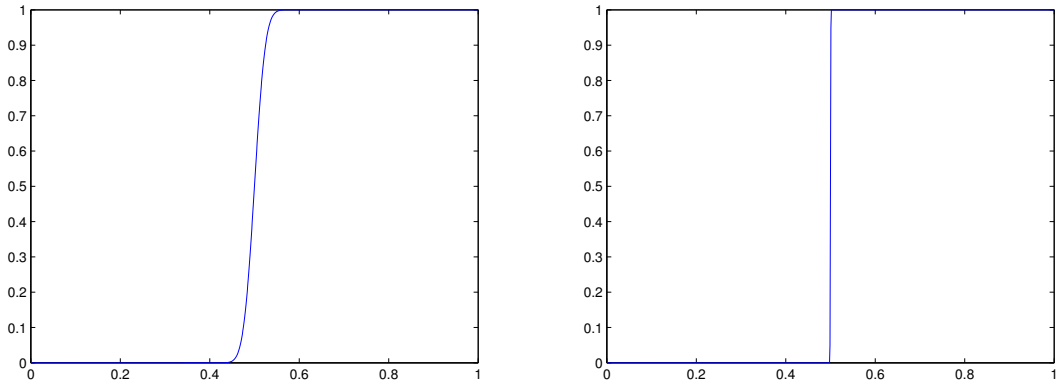


Figure 6.3: Probability density function when $n = 1000$



(a) $n = 1000$ (b) $n = 1000000$

Figure 6.4: Cumulative density function

6.3 DDM Miner: A Web-Based Information Retrieval System for Multimedia Big Data

Multimedia Information Retrieval (MIR), i.e., identification and labelling of multimedia data, is an important task in multimedia research, especially for multimedia big data. This dissertation presents an MIR system, called DDM Miner, designed and developed in the Data Mining, Database & Multimedia (DDM) Research Group. DDM Minter enables various feature extraction, model integration, and big data techniques to process and manage multimedia big data, integrates and fuses the scores obtained from different extraction and representation models, and generates semantic labels for given images for information retrieval. It uses the Spark platform and high performance NoSQL (Not Only SQL) databases to enable schema-less data models and distributed computation to speed up information retrieval.

6.3.1 Multimedia Information Retrieval

Bridging the gap between multimedia data and its semantic meaning effectively is crucial in MIR, especially in multimedia big data. It is well-acknowledged that there are huge variations in multimedia big data, and it is hard to find a general method that classifies all kinds of multimedia data with high accuracies. This is also due to the fact that various features capture disparate characteristics of the data, and thus contribute to semantics mining differently. On the other hand, different classifiers can handle heterogeneous multimedia data in various conditions. Therefore, feature fusion and classifier ensemble techniques can be applied to utilize and integrate the features and scores obtained from different models to analyse the semantic meanings of the multimedia data [48, 49].

Recently, deep learning has been used and greatly improved the retrieval accuracy. For example, ResNet [215] can achieve a 4.8% top-5 error rate and a 20.1% top-1 error rate for the ImageNet classification task. However, training a deep neural network (DNN) requires a sufficiently large training dataset and a great amount of computational resources. Furthermore, DNNs trained on one kind of multimedia data might not work well for data from different sources since DNNs cannot capture all the semantics in the data that are required for information retrieval.

To solve the above issues and enhance the retrieval performance of the deep learning techniques, we propose to utilize both the high-level features obtained from DNNs and the conventional hand-crafted features such as Histogram of Oriented Gradient (HOG) features, Scale-invariant feature transform (SIFT) features, Haar-like features, and color space information [72]. In addition, the scores from different classifiers in-

cluding Support Vector Machine (SVM), Logistic Regression (LR), Random Forrest (RF), etc. will be integrated to achieve the best retrieval performance.

6.3.2 DDM Miner System

Our DDM Miner system is a novel distributed and web-based system, hosted on a Spark cluster as shown in Figure 3.3, that adopts a collaborative filtering approach [216]. The Spark cluster is a heterogeneous configuration consisting of four nodes (one master and three worker nodes), running Yarn and Mesos for cluster and data management. DDM Miner allows a user to upload the multimedia data of interest (e.g., an image) via the web interface (as shown in Figure 6.5) to mine its underlying semantics (e.g., concept labels/classes) for information retrieval. The users can upload their own images to the server and visually verify the proper submission of the input file. The system provides various deep learning and ensemble based models to fuse low-level features from different layers for semantics mining. Such selection enables the user to explore the predictability and usefulness of each feature/classifier combination. That is, once the upload is complete, the user can review the query type, select the model(s) to generate the corresponding features, and decide the score integration technique to determine the labels of the uploaded data. If applicable, the parameter setup of the selected model can be modified in the right panel.

The DDM Miner system will detect the proper replication of the instances and schedule the feature extraction and model execution on the respective nodes to mine the concept labels. The proposed system can also generate conventional low-level features that can capture some distinguished characteristics of the images. The classification result of a sample image is shown in Figure 6.6, where the uploaded image

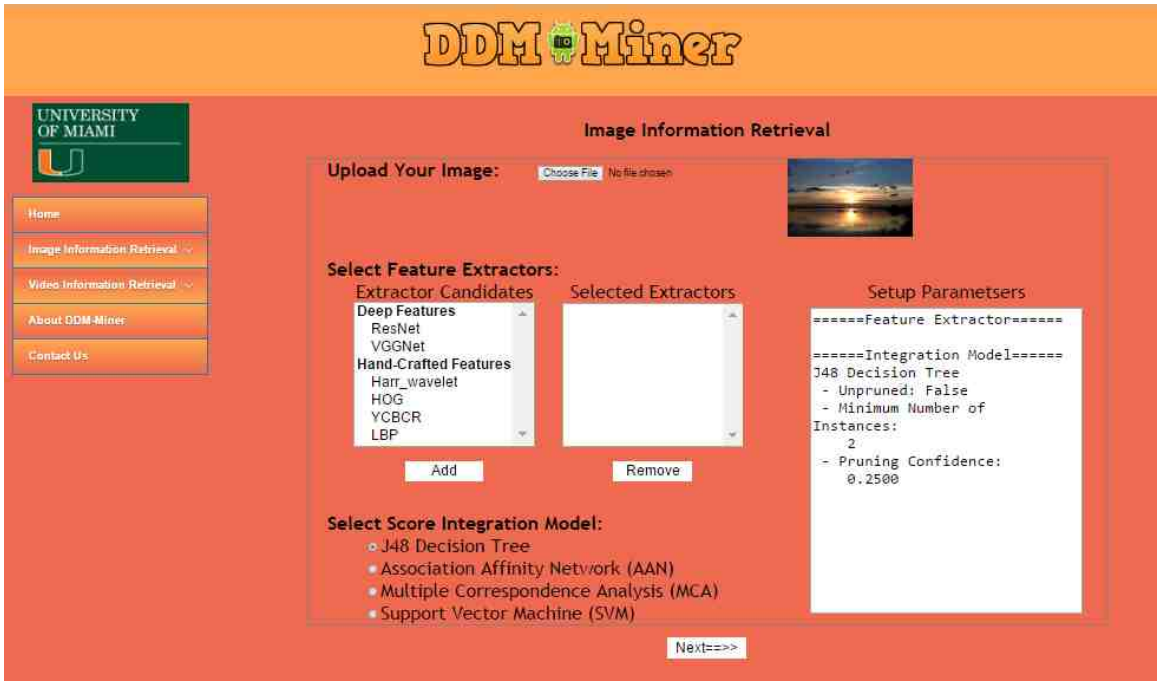


Figure 6.5: The image classification interface

is in the middle and the top 5 concepts (labels/classes) are displayed on the bottom along with their representative images. The data, related images, features, and labels are stored in a NoSQL database under Cassandra. The semantics information of the different images will be consolidated in a Score Integration module which uses Association Rule Mining (ARM) or Multiple Correspondence Analysis (MCA) to integrate the scores of similar images [163]. Final labels will be transformed in an XML wrapper and passed to the web-based GUI where it will be displayed to the user.

Compared to similar online portals, our DDM Miner system uses a big data environment (Spark) and assembles multiple feature and fusion technique choices. Its interface is user friendly and interactive, and allows the option for enhancement training.

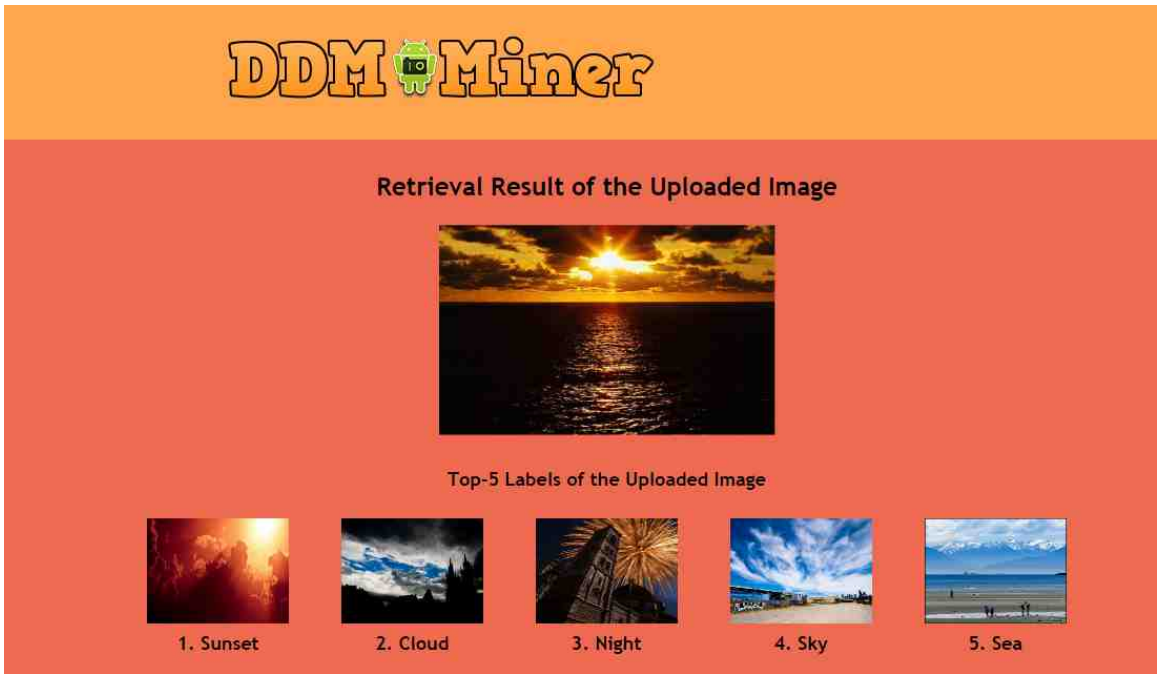


Figure 6.6: The classification result for a sample image

CHAPTER 7

Conclusions and Future Work

This chapter concludes all the previous chapters, develops the direction for future research, and introduces latest research progress on deep learning.

7.1 Conclusion

Classification of imbalanced data is an important research problem as lots of real-world datasets have skewed class distributions in which the majority of the examples (or instances) belong to some classes and far fewer instances belong to others. While in many applications, the minority instances actually represent the concept of interest (e.g., fraud in banking operations, abnormal cell in medical data, etc.), a classifier induced from an imbalanced dataset is more likely to be biased towards the majority classes and show very poor classification accuracy on the minority classes. Despite extensive research efforts, imbalanced data classification remains one of the most challenging problems in data mining and machine learning, especially for multimedia data.

In this dissertation, an extended CNN framework is proposed by integrating it with a bootstrapping strategy. During the bootstrapping process, a set of pseudo balanced training batches are generated based on the properties of the dataset and

fed into the CNN for classification. Using the TRECVID, KTH, and UCF11 datasets, the experimental results demonstrate the effectiveness of our framework in classifying multimedia data with a highly skewed data distribution. In addition, different from many existing studies in deep learning that take raw media data as the input, it is shown that our extended CNN framework can work effectively on low-level features, which greatly shortens the required training time in deep learning.

Next, a novel concept correlation analysis strategy framework using the correlation between the retrieval scores and labels is proposed to enhance rare class/concept mining. The experimental results on TRECVID dataset clearly show the effectiveness of the proposed framework and how it can successfully enhance the prediction scores of the chosen rare concepts.

For the third component, a novel classifier ensemble framework is proposed to fuse the classification results generated from different classifiers using different features. As a proof-of-concept, the proposed framework is applied to categorize human actions in videos. Experimental results on the KTH and UCF11 datasets show that the proposed framework is capable of taking advantages of different classifiers and outperforms some existing state-of-the-art approaches.

Two applications in the field of text data analysis prove the effectiveness of the proposed framework for real-world tasks. The first one is for Amazon review dataset while the second one proposes a novel framework to detect the stance between the followers of the two dominant presidential candidates, i.e., Hillary and Trump. For our best knowledge, we are the first group that uses machine learning algorithms for stance analysis in election predictions. Furthermore, a web-based information retrieval system is built to better demonstrate our work.

All in all, a Spark-based big data processing environment is implemented for all the components. The proposed framework can handle big datasets as it is integrated with Spark.

7.2 Future Work

On the basis of the current solutions and experimental results, several future research directions are identified and proposed in the following subsections.

7.2.1 Deep Learning on Quality of Feedback and Sub-category Classification of Reviews

In the existing framework, only text data in the Amazon review dataset are used. Meanwhile, based on user metadata, review metadata, and the average sentiment score of the reviews, a latent class profile can be created while multiple factor analysis could be applied on the mixed feature dataset. Then, hierarchical clustering can provide more information with the subgroup analysis of people reviewing for a particular product/service.

To identify fake reviews, we can consider each keyword in a review as a concept and find the recurring itemset of a particular individual. It was observed that incentivized reviews have consistently positive or negative sentiments on their accounts. For non-incentivized comments, individuals who endear a product tend to use different keywords as compared to those who dislike it. If we are able to find multiple instances of an account continuously using similar keywords, we have a high probability of identifying fake reviews in the future framework. Due to the imbalance

characteristic of all consumer feedback data, part of future experiments will be conducted to an experiment where we use bots to create fake comments on Amazon. Bots will be learnt to comment both good and bad about the products, and then those comments would be used in future experiments. This will help to develop the benchmarks of fake reviews and aid in decisions about other users.

Finally, after filtering all low-quality and fake reviews, each review will be split into sentences which will be labelled in one of the sub-categories using the current framework in Section 6.1. For each sub-category (e.g., service), we pick some service-related words and use the word2vec [200, 201] model to find similar words to build a word set for the service sub-category. It was observed that the keywords and their affinities are highly domain and topic specific, i.e., user behaviour is heterogeneous when spanned over different categories [196]. To overcome the challenge of domain invariance, we previously have successfully used the log odds ratio approach in Section 6.2. To further extend the existing framework on multiple categories and find most distinct keywords for the sub-category classification problem, all the sentences will be transformed into feature vectors for classification and a rating will be generated for each sub-category. Furthermore, if a review covers several sub-categories, it will be more possible to be considered as a high quality review.

Figure 7.1 depicts future work of the review quality and sub-category classification. It consists of four parts: input (blue), review rating (green), review data formatting (orange), and sentence-level review analysis (red). In general, the future framework collects the review data, analyses them, and generates the quality of feedback and the sub-category classification results of reviews.

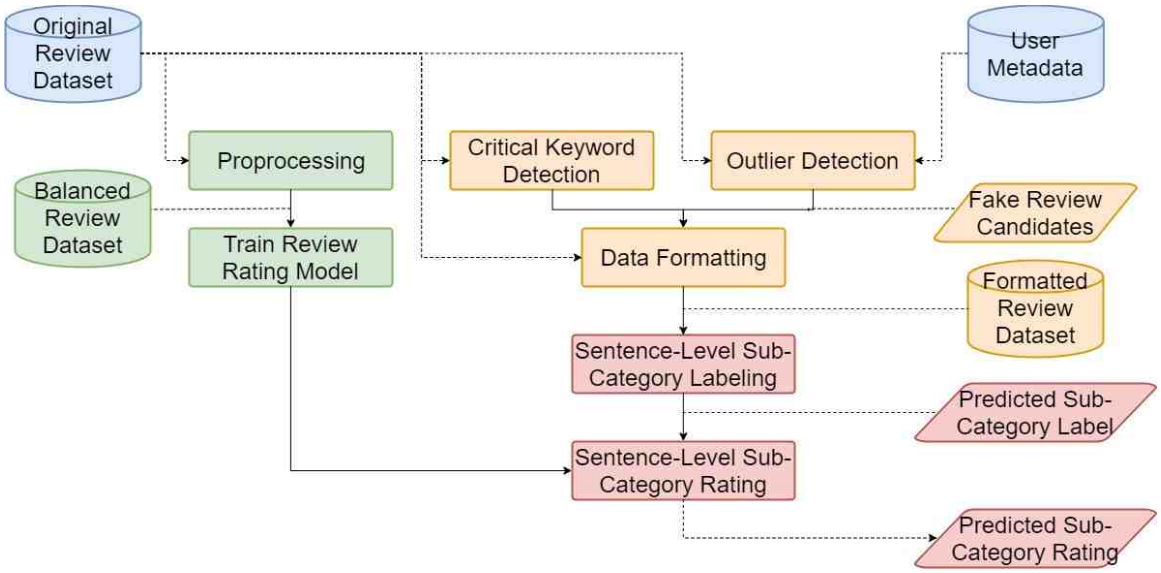


Figure 7.1: Our future framework on Amazon review dataset

As discussed before, given an Amazon review dataset or a review (text) online, we will first judge whether the review is low-quality or high-quality as well as from a bad buyer or nice buyer. Then, the estimated overall rating and the rating for each category will be generated by our framework. The preliminary expected results can be found in Table 7.1.

Table 7.1: Sample results of future work on Amazon review dataset

(a) Sample results of fake review detection

Attribute	Quality of feedback
Fake or not	0.13($Negative$)
Review quality	0.84($High$)

(b) Sample results of sub-category classification

Sub-category	Prediction	
Pricing	2 stars	
Quality	3 stars	
Delivery and Packaging	1 star	
Parts	Cable	4 stars
	Mouse	N/A
	Screen	1 star
Overall rating	3 stars	

From the Table 7.1a, we consider the review is unlikely a fake review based on our model. Furthermore, since the review covers several sub-categories, which means the reviewer fully considered many factors, it is labelled as a high quality review. Such

a review can be highlighted when Amazon users browse the product's webpage. In Table 7.1b, our model gives the details of the reviewer's attitude on each sub-category. It is clear that the reviewer is quite satisfied about the packaging while think she/he received it too late. The average rating of each sub-category of the product will be shown on the product's webpage to help Amazon users know the details of the product and also help the merchant improve the shipment process and other weak points.

7.2.2 Improving Stance Analysis with Advanced Information and Multi-modality Training

For stance analysis, an important insight is to observe if the accuracy of the clustering methods would be affected if we use real accounts with unknown predilections. This would help us also identify and evaluate the undecided voters. Currently, it is out of the scope to assert the ground truth for accounts having relative unknowns, but an extension of this framework will collect and extend the dataset with hand labelled real accounts and re-evaluate the stance.

Besides, other information in tweets including the resources, retweets, favorites, etc. would be also considered for better stance detection and bot classification in the future. In order to handle billions of tweets and get more data of supporters, more twitter accounts with labels must be included. One possible way is to get twitter accounts following one of the candidates, for example Hillary supporters' accounts. We can add those accounts to Hillary supporters' dataset after removing outliers.

It is noticeable that Hillary Clinton and Donald Trump supporters tend to post different kinds of data such as images and videos from various domains. Since most

of the current deep learning models only focus a single modality leading to an impoverished model of the world, research efforts are now also paid on cross-modality structures which may yield a huge step forward in deep learning. Some teams have been trying to use a multi-model architecture with modality-nets for multi-modality training which can simultaneously learn multiple tasks from various domains [217]. For different data types, different small modality-nets will be used to deal with inputs and outputs, e.g., image, language, and audio. For the "body" neural networks, we may use convolutional layers, an attention mechanism, and sparsely-gated layers. All in all, an unified deep learning model could be trained in the future to deal with multimodal inputs in stance analysis.

7.2.3 DDM Miner on a Big Data Processing Platform

While many popular big data systems adopted Spark, the designed Spark-based big data processing system can be further upgraded to a data processing platform architecture with Spark, Mesos, Akka, Cassandra, and Kafka (SMACK). Furthermore, Spark can be deployed on different kinds of resource management systems such as Apache Mesos^{TM} [103] which is a cluster resource management system with efficient resource isolation and sharing across distribute applications. It has also been shown that traditional Relational Database Management Systems (RDBMSs) are not able to meet the big data management needs and have become the bottleneck of a big data processing platform. New database technologies like NoSQL are the solution. Among them, Apache Cassandra^{TM} [97] is a distributed, highly available database designed to handle large amounts of data across multiple data centers. Using these techniques, an efficient multimedia big data mining system can be built to improve DDM Miner

on a big data processing platform for multimedia semantic concept retrieval. As part of the existing work, the use of these big data techniques will be explored in the future for multimedia big data analysis.

Furthermore, to take advantages of modern search engines, we plan to crawl and grab similar multimedia data online and feed them into our models to help semantics mining (e.g., labelling). Therefore, DDM Miner will be able to illustrate the performance of feature fusion and classifier ensemble for general MIR, and as the first step to experiment which sets of features and/or classifiers are more suitable on the given multimedia data. Here, we use the example of images to illustrate our future system. The web-based system initiates a remote procedure call to the Spark master node instance that initiates a web crawler using Google Image Search Rest API. This crawler will search for the top ten visually similar images of the uploaded data. The inflated dataset will then be streamed into the cluster using SPARK streaming engine and replicated across all nodes. Figure 7.2 illustrates its flowchart. As part of the future work, the functionalities of all the components of the proposed framework will be integrated (whenever applicable) into this web-cased DDM Miner system.

7.3 Latest Developments in Deep Learning

To date, a number of frameworks have been proposed by implementing the latest deep learning techniques and have shown promising results across a wide variety of domains including NLP (e.g., sentence classification, translation, etc.), visual data processing (e.g., computer vision, multimedia data analysis, etc.), speech and audio processing (e.g., enhancement, recognition, etc.), social network analysis, and healthcare. Other than all the aforementioned applications, deep learning algorithms are

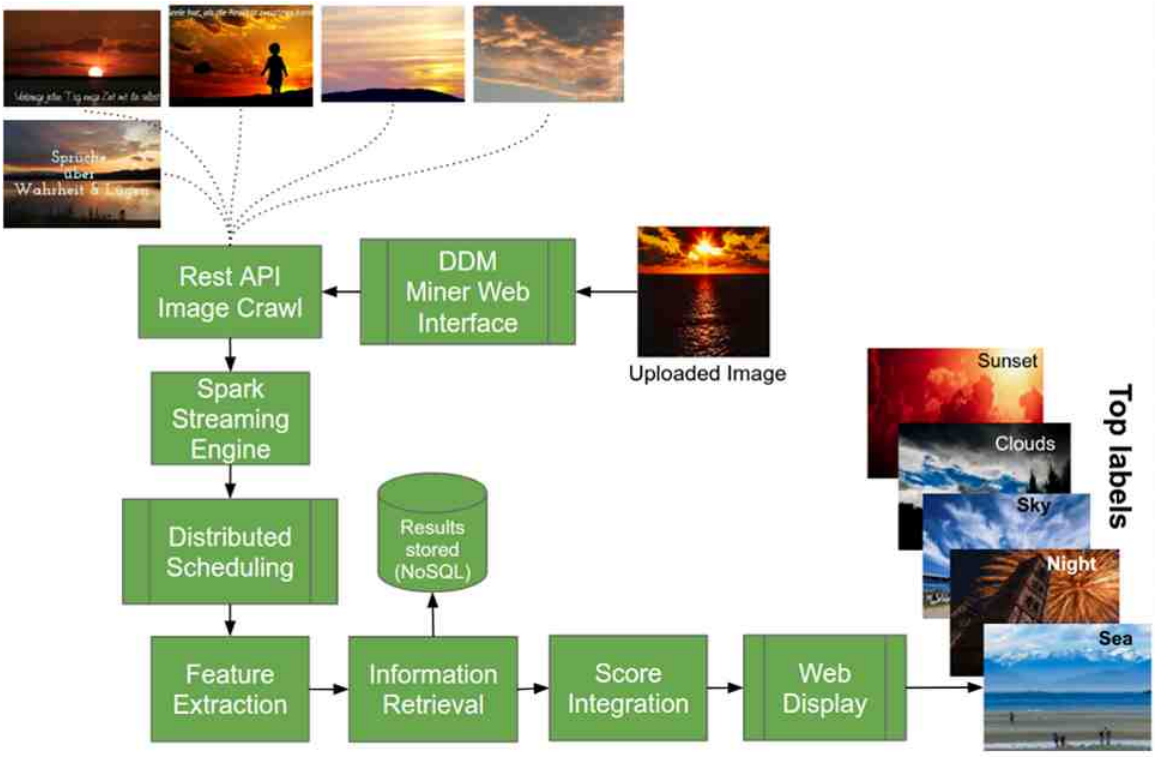


Figure 7.2: Flowchart of the future DDM Miner system

also applied to information retrieval, robotics, transportation prediction, autonomous driving, data compression, outlier detection, biomedicine, disaster management, etc. Please note that deep learning has shown its capability to be leveraged in various applications and only some of the selected applications are introduced here.

With the enormous amounts of data collected worldwide every day, the most non-trivial tasks in training a deep neural network is the training part apparently [76]. While GPU are well-known for large-scale matrices computing in network architectures on a single machine, a number of distributed deep learning frameworks have been developed to speed up the training part in deep learning [48, 218, 219]. Some popular deep learning frameworks like TensorFlow [220] also start to support parallel training on distributed systems. While huge amounts of data come without labels or with noisy labels, some researchers change their interests to improve noise robustness

of the training modules. They use unsupervised or semi-supervised deep learning to train deep neural networks subject to class-dependent label noise, as well as automatically remove outliers scattering among practical data collections [221, 222]. Less than one year after the success of AlphaGo, DeepMind trained an update version, AlphaGo Zero, without using data from human games but stronger than any previous version which highlighted the importance of training using unlabelled data [223]. Just two months later, their newest version Alpha Zero with a multi-skilled architecture achieved a superhuman level of play in both Shogi and Chess.

Researchers had paid lots of efforts using computer vision techniques to navigate through outdoor environments and plan around distant obstacles. With modern transportation prediction and vision-based navigation techniques, autonomous driving has gone from possible to inevitable. A large number of big companies and unicorn startups are working on self-driving automotive technologies including Google, Tesla, Aurora, Uber, etc. Back to 2008, Hadsell *et al.* used a relatively simply deep belief network with two convolutional layers and one max subsampling layer to extract deep features [224]. They used a self-supervised learning approach to achieve long-range vision in off-road terrain by training a classifier to discriminate the feature vectors. While autonomous driving technology is now more and more mature, a self-driving Uber car killed an Arizona pedestrian on March 19th, 2018. This shows that it still has a long way to go.

Another unintuitive deep learning application is data compression, especially lossless compression. Researchers train a special kind of neural networks, "autocoder", which has the same numbers of inputs and outputs. A simple autoencoder neural network uses unsupervised learning and applies backpropagation to minimize the dis-

tance between inputs and outputs, i.e., reconstruction error. It can be utilized for data compression if the number of hidden units is smaller than the number of inputs or outputs. The difference between an autoencoder and a general purpose compression tool like 7-zip is that the autoencoder is particularly trained for some data. A most recent lossless compression tool, DeepZip [225], is proposed by Kedar Tatwawadi, which consists of a RNN based probability estimator and an arithmetic coding block for large-scale data. The former one is trained to estimate the conditional probability distribution of a piece of data based on the rest of data, while the later one uses the estimated probabilities for coding and decoding.

Based on the observation that the inliers and the outliers can be separated by their reconstruction errors when the data are reconstructed from deep learning features, an autoencoder can also be utilized for outlier detection. Xia *et al.* [222] use the reconstruction errors of an autoencoder to automatically remove outliers from noisy data. They prove that it is an efficient tool for the unsupervised outlier removal task based on the difference between the nature of normal data and outliers. In the learning process of the autoencoder, discriminative information is progressively injected to make the inliers and the outliers more separable. They only minimize the reconstruction errors for the positive data rather than all the data to make the reconstruction errors more discriminative. By doing so, the reconstruction errors of the positive data are even smaller; while those of the outliers are not. Their idea was implemented in two steps iteratively and adaptively. First, the data are labelled as positive data or outliers according to their reconstruction errors, and then the network parameters are updated in the autoencoder by reducing the errors of the positive data, resulting in a more discriminative reconstruction.

Bibliography

[1] S.-C. Chen, "Multimedia databases and data management: a survey," *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, vol. 1, no. 1, pp. 1–11, 2010.

[2] M.-L. Shyu, S.-C. Chen, Q. Sun, and H. Yu, "Overview and future trends of multimedia research for content access and distribution," *International Journal of Semantic Computing*, vol. 1, no. 01, pp. 29–66, 2007.

[3] X. Huang, S.-C. Chen, M.-L. Shyu, and C. Zhang, "User concept pattern discovery using relevance feedback and multiple instance learning for content-based image retrieval," in *Proceedings of the Third International Workshop on Multimedia Data Mining, in conjunction with the 8th ACM International Conference on Knowledge Discovery & Data Mining*, July 2002, pp. 100–108.

[4] S.-C. Chen, M.-L. Shyu, and C. Zhang, "An intelligent framework for spatio-temporal vehicle tracking," in *Proceedings of the 4th IEEE International Conference on Intelligent Transportation Systems*, August 2001, pp. 213–218.

[5] M.-L. Shyu, C. Haruechaiyasak, S.-C. Chen, and N. Zhao, "Collaborative filtering by mining association rules from user access sequences," in *Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration*, April 2005, pp. 128–135.

[6] S.-C. Chen, S. Rubin, M.-L. Shyu, and C. Zhang, "A dynamic user concept pattern learning framework for content-based image retrieval," *IEEE Transactions on Systems, Man, and Cybernetics: Part C*, vol. 36, pp. 489–495, November 2006.

[7] S.-C. Chen, M.-L. Shyu, C. Zhang, and M. Chen, "A multimodal data mining framework for soccer goal detection based on decision tree logic," *International Journal of Computer Applications in Technology*, vol. 27, pp. 312–323, 2006.

[8] C. Haruechaiyasak, M.-L. Shyu, and S.-C. Chen, "Web document classification based on fuzzy association," in *Proceedings 26th Annual International Computer Software and Applications*, 2002, pp. 487–492.

[9] S.-C. Chen, M.-L. Shyu, C. Zhang, and J. Strickrott, "Multimedia data mining for traffic video sequences," in *Proceedings of the Second International Conference on Multimedia Data Mining*. Springer-Verlag, 2001, pp. 78–86.

[10] M. Chen, S.-C. Chen, and M.-L. Shyu, "Hierarchical temporal association mining for video event detection in video databases," in *Proceedings of the IEEE International Workshop on Multimedia Databases and Data Management (MDDM07)*, April 2007, pp. 137–145.

[11] L. Lin, G. Ravitz, M.-L. Shyu, and S.-C. Chen, "Correlation-based video semantic concept detection using multiple correspondence analysis," in *Proceedings of the IEEE International Symposium on Multimedia*, December 2008, pp. 316–321.

[12] C. Zhang, X. Chen, M. Chen, S.-C. Chen, and M.-L. Shyu, "A multiple instance learning approach for content based image retrieval using one-class support vector machine," in *Proceedings of the IEEE International on Multimedia & Expo*, July 2005, pp. 1142–1145.

[13] S.-C. Chen, M.-L. Shyu, W. Liao, and C. Zhang, "Scene change detection by audio and video clues," in *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 365–368.

[14] S.-C. Chen, S. Sista, M.-L. Shyu, and R. L. Kashyap, "An indexing and searching structure for multimedia database systems," in *Proceedings of the IS&T/SPIE conference on Storage and Retrieval for Media Databases*, January 2000, pp. 262–270.

[15] S.-C. Chen, M.-L. Shyu, C. Zhang, L. Luo, and M. Chen, "Detection of soccer goal shots using joint multimedia features and classification rules," *MDM/KDD*, vol. 3, 2003.

[16] S.-T. Li and S.-C. Chen, "Function approximation using robust wavelet neural networks," in *Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings. 14th IEEE International Conference on*. IEEE, 2002, pp. 483–488.

[17] K. Saleem, S. Luis, Y. Deng, S.-C. Chen, V. Hristidis, and T. Li, "Towards a business continuity information network for rapid disaster recovery," in *Proceedings of the 2008 international conference on Digital government research*. Digital Government Society of North America, 2008, pp. 107–116.

[18] C. Zhang, S.-C. Chen, M.-L. Shyu, and S. Peeta, "Adaptive background learning for vehicle detection and spatio-temporal tracking," in *Proceedings of the Fourth IEEE Pacific-Rim Conference On Multimedia*, December 2003, pp. 1–5.

[19] C. Zhang, S.-C. Chen, and M.-L. Shyu, "Pixso: a system for video shot detection," in *Proceedings of the Fourth IEEE Pacific-Rim Conference On Multimedia*, 2003, pp. 1–5.

[20] S.-C. Chen, M.-L. Shyu, S. Peeta, and C. Zhang, "Spatio-temporal vehicle tracking using unsupervised learning-based segmentation and object tracking," *IEEE Robotics and Automation Magazine, Special Issue on Robotic Technologies Applied to Intelligent Transportation Systems*, vol. 12, no. 1, pp. 50–58, 2005.

[21] T. Meng, A. T. Soliman, M.-L. Shyu, Y. Yang, S.-C. Chen, S. Iyengar, J. S. Yordy, and P. Iyengar, "Wavelet analysis in current cancer genome research: a survey," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 10, no. 6, pp. 1442–14 359, 2013.

[22] S.-C. Chen, M.-L. Shyu, and N. Zhao, "An enhanced query model for soccer video retrieval using temporal relationships," in *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on.* IEEE, 2005, pp. 1133–1134.

[23] C. Zhang, S.-C. Chen, and M.-L. Shyu, "Multiple object retrieval for image databases using multiple instance learning and relevance feedback." in *ICME*, vol. 4, 2004, pp. 775–778.

[24] K. Wickramaratna, M. Chen, S.-C. Chen, and M.-L. Shyu, "Neural network based framework for goal event detection in soccer videos," 2005.

[25] M.-L. Shyu, S.-C. Chen, M. Chen, and C. Zhang, "Affinity relation discovery in image database clustering and content-based retrieval," in *Proceedings of the 12th annual ACM international conference on Multimedia.* ACM, 2004, pp. 372–375.

[26] X. Li, S.-C. Chen, M.-L. Shyu, and B. Furht, "Image retrieval by color, texture, and spatial information," in *Proceedings of the 8th International Conference on Distributed Multimedia Systems*, September 2002, pp. 152–159.

[27] S.-C. Chen, M.-L. Shyu, and C. Zhang, "Innovative shot boundary detection for video indexing," in *Video Data Management and Information Retrieval*, S. Deb, Ed. Idea Group Publishing, 2005, pp. 217–236.

[28] X. Chen, C. Zhang, S.-C. Chen, and S. Rubin, "A human-centered multiple instance learning framework for semantic video retrieval," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 39, no. 2, pp. 228–233, 2009.

[29] N. Rishe, J. Yuan, R. Athauda, S.-C. Chen, X. Lu, X. Ma, A. Vaschillo, A. Shaposhnikov, and D. Vasilevsky, "Semantic access: semantic interface for querying databases," in *VLDB*, 2000, pp. 591–594.

[30] M.-L. Shyu, K. Sarinnapakorn, I. Kuruppu-Appuhamilage, S.-C. Chen, L. Chang, and T. Goldring, "Handling nominal features in anomaly intrusion

detection problems," in *15th International Workshop on Research Issues in Data Engineering: Stream Data Mining and Applications (RIDE-SDMA 2005)*, 2005, pp. 55–62.

[31] C. Chen, Q. Zhu, L. Lin, and M.-L. Shyu, "Web media semantic concept retrieval via tag removal and model fusion," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, pp. 61:1–61:22, October 2013.

[32] L. Zheng, C. Shen, L. Tang, T. Li, S. Luis, S.-C. Chen, and V. Hristidis, "Using data mining techniques to address critical information exchange needs in disaster affected public-private networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 125–134.

[33] Q. Zhu, L. Lin, M.-L. Shyu, and S.-C. Chen, "Effective supervised discretization for classification based on correlation maximization," in *Proceedings of the IEEE International Conference on Information Reuse and Integration*, 2011, pp. 390–395.

[34] S.-C. Chen, M.-L. Shyu, and R. Kashyap, "Augmented transition network as a semantic model for video data," *International Journal of Networking and Information Systems*, vol. 3, no. 1, pp. 9–25, 2000.

[35] S.-C. Chen, M. Chen, N. Zhao, S. Hamid, K. Chatterjee, and M. Armella, "Florida public hurricane loss model: Research in multi-disciplinary system integration assisting government policy making," *Government Information Quarterly*, vol. 26, no. 2, pp. 285–294, 2009.

[36] L. Zheng, C. Shen, L. Tang, C. Zeng, T. Li, S. Luis, and S.-C. Chen, "Data mining meets the needs of disaster information management," *IEEE Transactions on Human-Machine Systems*, vol. 43, pp. 451–464, 2013.

[37] L. Zheng, C. Shen, L. Tang, T. Li, S. Luis, and S.-C. Chen, "Applying data mining techniques to address disaster information management challenges on mobile devices," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 283–291.

[38] M.-L. Shyu, S.-C. Chen, and C. Haruechaiyasak, "Mining user access behavior on the www," in *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, vol. 3. IEEE, 2001, pp. 1717–1722.

[39] L. Lin, M.-L. Shyu, G. Ravitz, and S.-C. Chen, "Video semantic concept detection via associative classification," in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*. IEEE, 2009, pp. 418–421.

[40] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 157–166.

[41] Y. Sun, T. Osawa, K. Sudo, Y. Taniguchi, H. Li, Y. Guan, and L. Liu, "Trecvid 2013 semantic video concept detection by ntt-mddut," *TRECVID 2013*, pp. 26–28, 2013.

[42] C. Snoek, K. Van De Sande, D. Fontijne, A. Habibian, M. Jain, S. Kordumova, Z. Li, M. Mazloom, S. Pintea, R. Tao *et al.*, "Mediamill at trecvid 2013: Searching concepts, objects, instances and events in video," in *NIST TRECVID Workshop*, 2013.

[43] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *IEEE conference on Computer Vision and Pattern Recognition*. OH, USA: IEEE Computer Society, 2014, pp. 1725–1732.

[44] S.-C. Chen, S. Sista, M.-L. Shyu, and R. Kashyap, "Augmented transition networks as video browsing models for multimedia databases and multimedia information systems," in *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, 1999, pp. 175–182.

[45] X. Chen, C. Zhang, S.-C. Chen, and M. Chen, "A latent semantic indexing based method for solving multiple instance learning problem in region-based image retrieval," in *Proceedings of the 7th IEEE International Symposium on Multimedia*, Dec 2005, pp. 37–44.

[46] M. L. Shyu, Z. Xie, M. Chen, and S. C. Chen, "Video semantic event/concept detection using a subspace-based multimedia data mining framework," *IEEE Transactions on Multimedia*, vol. 10, no. 2, pp. 252–259, Feb 2008.

[47] Y. Yan, M.-L. Shyu, and Q. Zhu, "Supporting semantic concept retrieval with negative correlations in a multimedia big data mining system," *International Journal of Semantic Computing*, vol. 10, pp. 247–268, 2016.

[48] Y. Yan, Q. Zhu, M.-L. Shyu, and S.-C. Chen, "A classifier ensemble framework for multimedia big data classification," in *Proceedings of the IEEE 17th International Conference on Information Reuse and Integration (IRI)*, July 2016, pp. 615–622.

[49] Y. Yan, M. Chen, S. Sadiq, and M.-L. Shyu, "Efficient imbalanced multimedia concept retrieval by deep learning on spark clusters," *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, vol. 8, no. 1, pp. 1–20, 2017.

[50] E. A. Cherman, J. Metz, and M. C. Monard, "Incorporating label dependency into the binary relevance framework for multi-label classification," *Expert Systems with Applications*, vol. 39, no. 2, pp. 1647–1655, February 2011.

[51] S. Perera, *Instant MapReduce Patterns - Hadoop Essentials How-to.* Packt Publishing, May 2005.

[52] E. Gabarron, L. Fernandez-Luque, M. Armayones, and A. Y. Lau, "Identifying measures used for assessing quality of youtube videos with patient health information: A review of current literature," *Interactive Journal of Medical Research*, vol. 2, no. 1, March 2013.

[53] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, October 2006, pp. 321–330.

[54] Y. Yan, S. Pouyanfar, Y. Tao, H. Tian, M. P. Reyes, M.-L. Shyu, S.-C. Chen, W. Chen, T. Chen, and J. Chen, "Florida international university-university of miami trecvid 2017," 2017.

[55] Y. Yan, S. Pouyanfar, S. Guan, H. Tian, H.-Y. Ha, M.-L. Shyu, S.-C. Chen, W. Chen, T. Chen, and J. Chen, "Florida international university-university of miami trecvid 2016," 2016.

[56] Y. Yan, M. Gavidia, T. Sayed, H.-Y. Ha, M.-L. Shyu, S.-C. Chen, W. Chen, and T. Chen, "Florida international university-university of miami trecvid 2015," 2015.

[57] M. Gavidia, T. Sayed, Y. Yan, Q. Zhu, M.-L. Shyu, S.-C. Chen, H.-Y. Ha, M. Ma, W. Chen, and T. Chen, "Florida international university-university of miami trecvid 2014."

[58] B. Yin, N. Ruiz, F. Chen, and E. Ambikairajah, "Investigating speech features and automatic measurement of cognitive load," in *Multimedia Signal Processing, 2008 IEEE 10th Workshop on.* IEEE, 2008, pp. 988–993.

[59] M. Merler, B. Huang, L. Xie, G. Hua, and A. Natsev, "Semantic model vectors for complex video event recognition," *IEEE Transactions on Multimedia*, vol. 14, no. 1, pp. 88–101, February 2012.

[60] Q. Zhu, Z. Li, H. Wang, Y. Yang, and M.-L. Shyu, "Multimodal sparse linear integration for content-based item recommendation," in *Proceedings of the IEEE International Symposium on Multimedia*, 2013, pp. 187–194.

[61] R. P. Duin and D. M. Tax, "Experiments with classifier combining rules," in *International Workshop on Multiple Classifier Systems.* Springer, 2000, pp. 16–29.

[62] Y. Yan, Y. Liu, M.-L. Shyu, and M. Chen, "Utilizing concept correlations for effective imbalanced data classification," in *Proceedings of the IEEE 15th International Conference on Information Reuse and Integration*, Aug 2014, pp. 561–568.

[63] S.-C. Chen and R. Kashyap, "Temporal and spatial semantic models for multimedia presentations," in *Proceedings of the 1997 International Symposium on Multimedia Information Processing*, 1997, pp. 441–446.

[64] S.-C. Chen, M.-L. Shyu, C. Zhang, and R. L. Kashyap, "Identifying overlapped objects for video indexing and modeling in multimedia database systems," *International Journal on Artificial Intelligence Tools*, vol. 10, no. 4, pp. 715–734, 2001.

[65] X. Li, S.-C. Chen, M.-L. Shyu, and B. Furht, "An effective content-based visual image retrieval system," in *Proceedings of the Computer Software and Applications Conference*, 2002, pp. 914–919.

[66] D. Liu, Y. Yan, M.-L. Shyu, G. Zhao, and M. Chen, "Spatio-temporal analysis for human action detection and recognition in uncontrolled environments," *International Journal of Multimedia Data Engineering and Management*, vol. 6, no. 1, pp. 1–18, Jan. 2015.

[67] M.-L. Shyu, C. Haruechaiyasak, and S.-C. Chen, "Category cluster discovery from distributed www directories," *Information Sciences*, vol. 155, no. 3, pp. 181–197, 2003.

[68] Y. Yan, M. Chen, M.-L. Shyu, and S.-C. Chen, "Deep learning for imbalanced multimedia data classification," in *Proceedings of the 2015 IEEE International Symposium on Multimedia (ISM)*, December 2015, pp. 483–488.

[69] M. Buckland and F. Gey, "The relationship between recall and precision," *Journal of the American society for information science*, vol. 45, no. 1, p. 12, 1994.

[70] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Transactions on Signal and Information Processing*, vol. 3, pp. 1–29, 2014.

[71] S. Pouyanfar and S.-C. Chen, "Semantic event detection using ensemble deep learning," in *The IEEE International Symposium on Multimedia (IEEE ISM)*, CA, USA, 2016, pp. 203–208.

[72] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[73] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, Kerkyra, Greece, August 1999, pp. 1150–1157.

[74] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, pp. 1–21, 2015.

[75] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[76] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[77] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[78] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[79] M. I. Jordan, "Serial order: A parallel distributed processing approach," *Advances in Psychology*, vol. 121, pp. 471–495, 1986.

[80] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[81] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[82] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[83] "Alphago," https://deepmind.com/research/alphago, 2016, accessed April, 2017.

[84] J. X. Chen, "The evolution of computing: Alphago," *Computing in Science Engineering*, vol. 18, no. 4, pp. 4–7, July 2016.

[85] C.-S. Lee, M.-H. Wang, S.-J. Yen, T.-H. Wei, I.-C. Wu, P.-C. Chou, C.-H. Chou, M.-W. Wang, and T.-H. Yan, "Human vs. computer go: Review and prospect [discussion forum]," *IEEE Computational Intelligence Magazine*, vol. 11, no. 3, pp. 67–72, 2016.

[86] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 20–29, Jun. 2004.

[87] X. Y. Liu, J. Wu, and Z. H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, April 2009.

[88] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[89] M. Wasikowski and X.-w. Chen, "Combating the small sample class imbalance problem using feature selection," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1388–1400, 2010.

[90] C. Junfei, W. Qingfeng, and D. Huailin, "An empirical study on ensemble selection for class-imbalance data sets," in *Proceedings of the 5th International Conference on Computer Science Education*, Aug 2010, pp. 477–480.

[91] U. Bhowan, M. Johnston, and M. Zhang, "Developing new fitness functions in genetic programming for classification with unbalanced data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 406–421, April 2012.

[92] Q. Zhu, L. Lin, M.-L. Shyu, and S.-C. Chen, "Feature selection using correlation and reliability based scoring metric for video semantic detection," in *Proceedings of the Fourth IEEE International Conference on Semantic Computing*, 2010, pp. 462–469.

[93] S. Hukerikar, A. Tumma, A. Nikam, and V. Attar, "Skewboost: An algorithm for classifying imbalanced datasets," in *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on.* IEEE, 2011, pp. 46–52.

[94] S. Ertekin, J. Huang, L. Bottou, and L. Giles, "Learning on the border: active learning in imbalanced data classification," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management.* ACM, 2007, pp. 127–136.

[95] I. Jamali, M. Bazmara, and S. Jafari, "Feature selection in imbalance data sets," *International Journal of Computer Science Issues*, vol. 9, no. 3, pp. 42–45, 2012.

[96] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 80–89, 2004.

[97] Apache, "Cassandra," http://cassandra.apache.org/, accessed March, 2017.

[98] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1327452.1327492

[99] Apache, "Hadoop," http://hadoop.apache.org, accessed October, 2017.

[100] ——, "Spark," https://spark.apache.org, accessed October, 2017.

[101] U. Berkeley, "amplab," https://amplab.cs.berkeley.edu, accessed October, 2017.

[102] R. S. Xin, J. Rosen, M. Zaharia, M. J. Franklin, S. Shenker, and I. Stoica, "Shark: Sql and rich analytics at scale," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: ACM, 2013, pp. 13–24. [Online]. Available: http://doi.acm.org/10.1145/2463676.2465288

[103] Apache, "Mesos," http://mesos.apache.org/, accessed March, 2017.

[104] P. Westfall and K. S. Henning, *Understanding advanced statistical methods*. CRC Press, 2013.

[105] E. Wong, T. Wei, Y. Qi, and L. Zhao, "A crosstab-based statistical method for effective fault localization," in *Software Testing, Verification, and Validation, 2008 1st International Conference on*. IEEE, 2008, pp. 42–51.

[106] W. Bergsma, "A bias-correction for cramérs v and tschuprows t," *Journal of the Korean Statistical Society*, vol. 42, no. 3, pp. 323–328, 2013.

[107] L. A. Goodman and W. H. Kruskal, "Measures of association for cross classifications, iv: Simplification of asymptotic variances," *Journal of the American Statistical Association*, vol. 67, no. 338, pp. 415–421, 1972.

[108] J. Piantadosi, P. Howlett, and J. Boland, "Matching the grade correlation coefficient using a copula with maximum disorder," *Journal of Industrial and Management Optimization*, vol. 3, no. 2, p. 305, 2007.

[109] Y. Yan, M.-L. Shyu, and Q. Zhu, "Negative correlation discovery for big multimedia data semantic concept mining and retrieval," in *Proceedings of the 2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, February 2016, pp. 55–62.

[110] T. Meng, Y. Liu, M.-L. Shyu, Y. Yan, and C.-M. Shu, "Enhancing multimedia semantic concept mining and retrieval by incorporating negative correlations," in *Proceedings of the IEEE International Conference on Semantic Computing*, June 2014, pp. 28–35.

[111] Y.-G. Jiang, J. Wang, S.-F. Chang, and C.-W. Ngo, "Domain adaptive semantic diffusion for large scale context-based video annotation," in *Proceedings of the12th IEEE International Conference on Computer Vision*, Sept 2009, pp. 1420–1427.

[112] K. Pearson, "Notes on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, 1895.

[113] X. Wang and Q. Ji, "A hierarchical context model for event recognition in surveillance video," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 2561–2568.

[114] B. Ni, Y. Song, and M. Zhao, "Youtubeevent: On large-scale video event classification," in *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, November 2011, pp. 1516–1523.

[115] Y. T. Yang and C. T. Chiu, "Boosted multi-class object detection with parallel hardware implementation for real-time applications," in *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 7530–7534.

[116] D. T. J. Vreeswijk, C. G. M. Snoek, K. E. A. van de Sande, and A. W. M. Smeulders, "All vehicles are cars: Subclass preferences in container concepts," in *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*. New York, NY, USA: ACM, 2012, pp. 8:1–8:7.

[117] L. Jiang, S.-I. Yu, D. Meng, Y. Yang, T. Mitamura, and A. G. Hauptmann, "Fast and accurate content-based semantic search in 100m internet videos," in *Proceedings of the 23rd ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2015, pp. 49–58.

[118] T. Meng and M.-L. Shyu, "Concept-concept association information integration and multi-model collaboration for multimedia semantic concept detection," *Information Systems Frontiers*, pp. 1–13, 2013.

[119] S.-C. Chen, A. Ghafoor, and R. L. Kashyap, *Semantic Models for Multimedia Database Searching and Browsing*. Norwell, MA, USA: Kluwer Academic Publishers, 2000.

[120] M.-L. Shyu, S.-C. Chen, M. Chen, and C. Zhang, "A unified framework for image database clustering and content-based retrieval," in *Proceedings of the 2Nd ACM International Workshop on Multimedia Databases*, ser. MMDB '04. New York, NY, USA: ACM, 2004, pp. 19–27.

[121] M.-L. Shyu, S.-C. Chen, M. Chen, C. Zhang, and K. Sarinnapakorn, "Image database retrieval utilizing affinity relationships," in *Proceedings of the 1st ACM International Workshop on Multimedia Databases*, ser. MMDB '03. New York, NY, USA: ACM, 2003, pp. 78–85.

[122] M.-L. Shyu, T. Quirino, Z. Xie, S.-C. Chen, and L. Chang, "Network intrusion detection through adaptive sub-eigenspace modeling in multiagent systems," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 2, no. 3, Sep. 2007.

[123] E. A. Perez, V. F. Mota, L. M. Maciel, D. Sad, and M. B. Vieira, "Combining gradient histograms using orientation tensors for human action recognition," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 3460–3463.

[124] Y. Yan and M.-L. Shyu, "Enhancing rare class mining in multimedia big data by concept correlation," in *Proceedings of the 2016 IEEE International Symposium on Multimedia (ISM)*, December 2016, pp. 281–286.

[125] Y. Yan, S. Pouyanfar, H. Tian, S. Guan, H.-Y. Ha, S.-C. Chen, M.-L. Shyu, and S. Hamid, "Domain knowledge assisted data processing for florida public hurricane loss model," in *Proceedings of the 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI)*. IEEE, 2016, pp. 441–447.

[126] H. Tian, H.-Y. Ha, S. Pouyanfar, Y. Yan, S. Guan, S.-C. Chen, M.-L. Shyu, and S. Hamid, "A scalable and automatic validation process for florida public hurricane loss model," in *Proceedings of the IEEE 17th International Conference on Information Reuse and Integration (IRI)*. IEEE, July 2016, pp. 324–331.

[127] Q. Zhu and M.-L. Shyu, "Sparse linear integration of content and context modalities for semantic concept retrieval," *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 2, pp. 152–160, June 2015.

[128] R. P. Duin, "The combining classifier: to train or not to train?" in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2. IEEE, 2002, pp. 765–770.

[129] A. B. Ashfaq, M. Javed, S. A. Khayam, and H. Radha, "An information-theoretic combining method for multi-classifier anomaly detection systems," in *Communications (ICC), 2010 IEEE international conference on*. IEEE, 2010, pp. 1–5.

[130] T. Meng, M.-L. Shyu, and L. Lin, "Multimodal information integration and fusion for histology image classification," in *Multimedia Data Engineering Applications and Processing*. IGI Global, 2013, pp. 35–50.

[131] N. Liu, E. Dellandréa, L. Chen, C. Zhu, Y. Zhang, C.-E. Bichot, S. Bres, and B. Tellez, "Multimodal recognition of visual concepts using histograms of textual concepts and selective weighted late fusion scheme," *Computer Vision and Image Understanding*, vol. 117, no. 5, pp. 493–512, 2013.

[132] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[133] L. I. Kuncheva, *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[134] T. Meng and M.-L. Shyu, "Leveraging concept association network for multimedia rare concept mining and retrieval," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, Melbourne, Australia, July 2012, pp. 860–865.

[135] L. Xu, A. Krzyzak, and C. Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE transactions on systems, man, and cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[136] A. Al-Ani and M. Deriche, "A new technique for combining multiple classifiers using the dempster-shafer theory of evidence," *Journal of Artificial Intelligence Research*, vol. 17, pp. 333–361, 2002.

[137] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng, "On optimization methods for deep learning," in *The 28th International Conference on Machine Learning*, Washington, USA, 2011, pp. 265–272.

[138] E. R. Kandel, "An introduction to the work of david hubel and torsten wiesel," *The Journal of physiology*, vol. 587, no. 12, pp. 2733–2741, 2009.

[139] J. Bouvrie, "Notes on convolutional neural networks," 2006.

[140] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision*. Springer, 2014, pp. 346–361.

[141] D. Verma, V. Maru *et al.*, "An efficient approach for color image retrieval using haar wavelet," in *Proceeding of International Conference on Methods and Models in Computer Science (ICM2CS)*. IEEE, 2009, pp. 1–5.

[142] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the hsv color space for image retrieval," in *International Conference on Image Processing*, vol. 2. IEEE, 2002.

[143] S. A. Chatzichristofis and Y. S. Boutalis, "Cedd: color and edge directivity descriptor: a compact descriptor for image indexing and retrieval," in *International Conference on Computer Vision Systems*. Springer, 2008, pp. 312–322.

[144] Yahoo, "Caffeonspark," https://github.com/yahoo/CaffeOnSpark, accessed October, 2017.

[145] Skymind, "Deeplearning4j," https://deeplearning4j.org/, accessed October, 2017.

[146] J. Liu, Y. Yang, and M. Shah, "Learning semantic visual vocabularies using diffusion distance," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 461–468.

[147] J. Liu, J. Luo, and M. Shah, "Recognizing realistic actions from videos "in the wild"," in *Computer vision and pattern recognition, 2009. CVPR 2009. IEEE conference on*. IEEE, 2009, pp. 1996–2003.

[148] I. Laptev, "On space-time interest points," *International journal of computer vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[149] V. F. Mota, E. d. A. Perez, L. M. Maciel, M. B. Vieira, and P. H. Gosselin, "A tensor motion descriptor based on histograms of gradients and optical flow," *Pattern Recognition Letters*, vol. 39, pp. 85–91, 2014.

[150] G. Awad, J. Fiscus, M. Michel, D. Joy, W. Kraaij, A. F. Smeaton, G. Quénot, M. Eskevich, R. Aly, and R. Ordelman, "Trecvid 2016: Evaluating video search, video event detection, localization, and hyperlinking," in *Proceedings of TRECVID*, vol. 2016, 2016.

[151] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and trecvid," in *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*. ACM, 2006, pp. 321–330.

[152] Q. Gu, L. Zhu, and Z. Cai, "Evaluation measures of the classification performance of imbalanced data sets," *Computational intelligence and intelligent systems*, pp. 461–471, 2009.

[153] N. Inoue and K. Shinoda, "A fast and accurate video semantic-indexing system using fast map adaptation and gmm supervectors," *IEEE Transactions on Multimedia*, vol. 14, no. 4, pp. 1196–1205, 2012.

[154] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[155] Y. Yan and M.-L. Shyu, "Correlation-assisted imbalance multimedia concept mining and retrieval," *International Journal of Semantic Computing*, vol. 11, no. 02, pp. 209–227, 2017.

[156] Y. Yan, M. Chen, S. Sadiq, and M.-L. Shyu, "Efficient imbalanced multimedia concept retrieval by deep learning on spark clusters," *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, vol. 8, no. 1, pp. 1–20, 2017.

[157] P. Mettes, D. C. Koelma, and C. G. M. Snoek, "The imagenet shuffle: Reorganized pre-training for video event detection," *CoRR*, vol. abs/1602.07119, 2016.

[158] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[159] S.-i. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Networks*, vol. 12, no. 6, pp. 783–789, 1999.

[160] K. P. Murphy, "Naive bayes classifiers," *University of British Columbia*, 2006.

[161] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[162] M. R. Segal, "Machine learning benchmarks and random forest regression," *Center for Bioinformatics & Molecular Biostatistics*, 2004.

[163] L. Lin and M.-L. Shyu, "Weighted association rule mining for video semantic detection," *International Journal of Multimedia Data Engineering and Management*, vol. 1, no. 1, pp. 37–54, 2010.

[164] L. Lin, G. Ravitz, M.-L. Shyu, and S.-C. Chen, "Effective feature space reduction with imbalanced data for semantic concept detection," in *Proceedings of the IEEE International on Sensor Networks, Ubiquitous, and Trustworthy Computing*, June 2008, pp. 262–269.

[165] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, W. Kraaij, A. F. Smeaton, G. Quéenot, and R. Ordelman, "Trecvid 2015 – an overview of the goals, tasks, data, evaluation mechanisms and metrics," in *Proceedings of TRECVID 2015*. NIST, USA, 2015.

[166] Q. Zhu, L. Lin, M.-L. Shyu, and D. Liu, "Utilizing context information to enhance content-based image classification," *International Journal of Multimedia Data Engineering and Management*, vol. 2, no. 3, pp. 34–51, 2011.

[167] L. Lin, C. Chen, M.-L. Shyu, and S.-C. Chen, "Weighted subspace filtering and ranking algorithms for video concept retrieval," *IEEE Multimedia*, vol. 18, no. 3, pp. 32–43, March 2011.

[168] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 1725–1732.

[169] Y.-G. Jiang, "Prediction scores on TRECVID 2010 data set," http://www.ee.columbia.edu/ln/dvmm/CU-VIREO374/, 2010, accessed September, 2017. [Online]. Available: http://www.ee.columbia.edu/ln/dvmm/CU-VIREO374/

[170] Y. Yan, Q. Zhu, M.-L. Shyu, and S.-C. Chen, "Classifier fusion by judgers on spark clusters for multimedia big data classification," in *Quality Software Through Reuse and Integration*. Springer, 2016, pp. 91–108.

[171] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3. IEEE, 2004, pp. 32–36.

[172] D. Liu and M.-L. Shyu, "Effective moving object detection and retrieval via integrating spatial-temporal multimedia information," in *Proceedings of the 2012 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2012, pp. 364–371.

[173] L.-C. Chen, J.-W. Hsieh, Y. Yan, and D.-Y. Chen, "Vehicle make and model recognition using sparse representation and symmetrical SURFs," *Pattern Recognition*, vol. 48, no. 6, pp. 1979 – 1998, 2015.

[174] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[175] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research*, vol. 11, no. Jan, pp. 19–60, 2010.

[176] K.-T. Chuang, J.-W. Hsieh, and Y. Yan, "Modeling and recognizing action contexts in persons using sparse representation," in *Advances in Intelligent Systems and Applications - Volume 2*, ser. Smart Innovation, Systems and Technologies, J.-S. Pan, C.-N. Yang, and C.-C. Lin, Eds. Springer Berlin Heidelberg, 2013, vol. 21, pp. 531–541.

[177] S.-Y. Wang, J.-W. Hsieh, Y. Yan, L.-C. Chen, and D.-y. Chen, "PLSA-based sparse representation for vehicle color classification," in *Proceedings of the IEEE 12th International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2015, pp. 1–6.

[178] J.-W. Hsieh, Y. Yan, L.-C. Chen, H.-F. Chiang, and H.-Y. Liu, "Object classification using PLSA and sparse representation," in *Information Technology and Applications in Outlying Islands (ITAOI), 2014 13th International Conference on*, May 2014.

[179] L.-C. Chen, J.-W. Hsieh, Y. Yan, and D.-Y. Chen, "Vehicle make and model recognition using sparse representation and symmetrical SURFs," in *Proceedings of the 16th IEEE International Conference on Intelligent Transportation Systems*, Oct 2013, pp. 1143–1148.

[180] J.-W. Hsieh, K.-T. Chuang, Y. Yan, and L.-C. Chen, "Sparse representation for recognizing object-to-object actions under occlusions," in *Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service*, ser. ICIMCS '13. New York, NY, USA: ACM, 2013, pp. 117–120.

[181] L.-C. Chen, J.-W. Hsieh, Y. Yan, and B.-Y. Wong, "Real-time vehicle make and model recognition from roads," in *Proceedings of the 12th International Conference on Information Technology and Applications in Outlying Islands*, May 2013, pp. 1033–1040.

[182] H.-F. Chiang, J.-W. Hsieh, C.-H. Chuang, K.-T. Chuang, and Y. Yan, "Modeling and recognizing action contexts in persons using sparse representation," *Journal of Visual Communication and Image Representation*, vol. 30, pp. 252 – 265, 2015.

[183] Y. Yan, J.-W. Hsieh, H.-F. Chiang, S.-C. Cheng, and D.-Y. Chen, "PLSA-based sparse representation for object classification," in *Proceedings of the 22nd International Conference on Pattern Recognition*, Aug 2014, pp. 1295–1300.

[184] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on.* IEEE, 2005, pp. 65–72.

[185] J. Yin and Y. Meng, "Human activity recognition in video using a hierarchical probabilistic latent model," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on.* IEEE, 2010, pp. 15–20.

[186] J. C. Niebles, C.-W. Chen, and L. Fei-Fei, "Modeling temporal structure of decomposable motion segments for activity classification," in *European conference on computer vision.* Springer, 2010, pp. 392–405.

[187] X.-g. Chen, J. Liu, and H. Liu, "Will scene information help realistic action recognition?" in *Intelligent Control and Automation (WCICA), 2012 10th World Congress on.* IEEE, 2012, pp. 4532–4535.

[188] V. F. Mota, J. I. Souza, A. d. A. Araújo, and M. B. Vieira, "Combining orientation tensors for human action recognition," in *Graphics, Patterns and Images (SIBGRAPI), 2013 26th SIBGRAPI-Conference on.* IEEE, 2013, pp. 328–333.

[189] P. Swietojanski, A. Ghoshal, and S. Renals, "Convolutional neural networks for distant speech recognition," *IEEE Signal Processing Letters*, vol. 21, no. 9, pp. 1120–1124, 2014.

[190] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *IEEE Geoscience and remote sensing letters*, vol. 11, no. 10, pp. 1797–1801, 2014.

[191] Q. Mao, M. Dong, Z. Huang, and Y. Zhan, "Learning salient features for speech emotion recognition using convolutional neural networks," *IEEE Transactions on Multimedia*, vol. 16, no. 8, pp. 2203–2213, 2014.

[192] U. C. Bureau, "Monthly & annual retail trade," https://www.census.gov/retail/mrts/www/data/pdf/ec_current.pdf, accessed August, 2017.

[193] Amazon, "Amazon product data," http://jmcauley.ucsd.edu/data/amazon/, accessed March 21st, 2018.

[194] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1422–1432.

[195] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy, "Hierarchical attention networks for document classification," in *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[196] E. Grave, T. Mikolov, A. Joulin, and P. Bojanowski, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017, pp. 427–431.

[197] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[198] J. Jagarlamudi, H. Daumé III, and R. Udupa, "Incorporating lexical priors into topic models," in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2012, pp. 204–213.

[199] G. E. Hinton *et al.*, "Learning distributed representations of concepts," in *Proceedings of the eighth annual conference of the cognitive science society*, vol. 1. Amherst, MA, 1986, p. 12.

[200] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[201] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[202] E. Kouloumpis, T. Wilson, and J. D. Moore, "Twitter sentiment analysis: The good the bad and the omg!" *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*, vol. 11, no. 538-541, p. 164, 2011.

[203] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of twitter data," in ., Association for Computational Linguistics. Proceedings of the Workshop on Languages in Social Media, 2011, pp. 30–38.

[204] J. Bollen, H. Mao, and A. Pepe, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena." in ., vol. 11. Barcelona, Spain: International AAAI Conference on Weblogs and Social Media, 2011, pp. 450–453.

[205] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining." in ., vol. 10. European Language Resources Association (ELRA), 2010.

[206] D. Robinson, "Text analysis of trump's tweets confirms he writes only the (angrier) android half," http://varianceexplained.org/r/trump-tweets, accessed May 7th, 2017.

[207] B. Stecanella, "Donald trump vs hillary clinton: sentiment analysis on twitter mentions," https://blog.monkeylearn.com/donald-trump-vs-hillary-clinton-sentiment-analysis-twitter-mentions, accessed May 7th, 2017.

[208] Y. Jia, "Trump vs hillary on twitter," http://yiyujia.blogspot.com/2016/09/trump-vs-hillary-on-twitter, accessed May 7th, 2017.

[209] S. Sadiq, Y. Yan, A. Taylor, M.-L. Shyu, S.-C. Chen, and D. Feaster, "AAFA: Associative affinity factor analysis for bot detection and stance classification in twitter," in *Proceedings of the IEEE 18th International Conference on Information Reuse and Integration (IRI)*. IEEE, 2017, pp. 356–365.

[210] S. Sadiq, Y. Tao, Y. Yan, and M.-L. Shyu, "Mining anomalies in medicare big data using patient rule induction method," in *Proceedings of the IEEE Third International Conference on Multimedia Big Data*. IEEE, 2017, pp. 185–192.

[211] Wikipedia, "List of donald trump presidential campaign endorsements, 2016," https://en.wikipedia.org/wiki/List_of_Donald_Trump_presidential_campaign_endorsements,_2016, accessed May 7th, 2017.

[212] S. Sadiq, Y. Yan, M.-L. Shyu, S.-C. Chen, and H. Ishwaran, "Enhancing multimedia imbalanced concept detection using vimp in random forests," in *Proceedings of the 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI)*, July 2016, pp. 601–608.

[213] M.-L. Shyu, S.-C. Chen, and R. Kashyap, "Generalized affinity-based association rule mining for multimedia database queries," *Knowledge and Information Systems (KAIS): An International Journal*, vol. 3, no. 3, pp. 319–337, August 2001.

[214] T. Li, F. Li, and X. Zhang, "Prediction of kinase-specific phosphorylation sites with sequence features by a log-odds ratio approach," *Proteins: Structure, Function, and Bioinformatics*, vol. 70, no. 2, pp. 404–414, 2008.

[215] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[216] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," *The Adaptive Web*, pp. 291–324, 2007.

[217] L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit, "One model to learn them all," *CoRR*, vol. abs/1706.05137, 2017. [Online]. Available: http://arxiv.org/abs/1706.05137

[218] E. Azarkhish, D. Rossi, I. Loi, and L. Benini, "Neurostream: Scalable and energy efficient deep learning with smart memory cubes," *CoRR*, vol. abs/1701.06420, 2017. [Online]. Available: http://arxiv.org/abs/1701.06420

[219] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

[220] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467

[221] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: a loss correction approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, HI, USA: IEEE, 2017, pp. 2233–2241.

[222] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun, "Learning discriminative reconstructions for unsupervised outlier removal," in *Proceedings of the IEEE International Conference on Computer Vision*. Santiago, Chile: IEEE, 2015, pp. 1511–1519.

[223] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 10 2017.

[224] R. Hadsell, A. Erkan, P. Sermanet, M. Scoffier, U. Muller, and Y. LeCun, "Deep belief net learning in a long-range vision system for autonomous off-road driving," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 628–633.

[225] K. Tatwawadi, "Deepzip: Lossless compression using recurrent networks," 2018, unpublished paper.