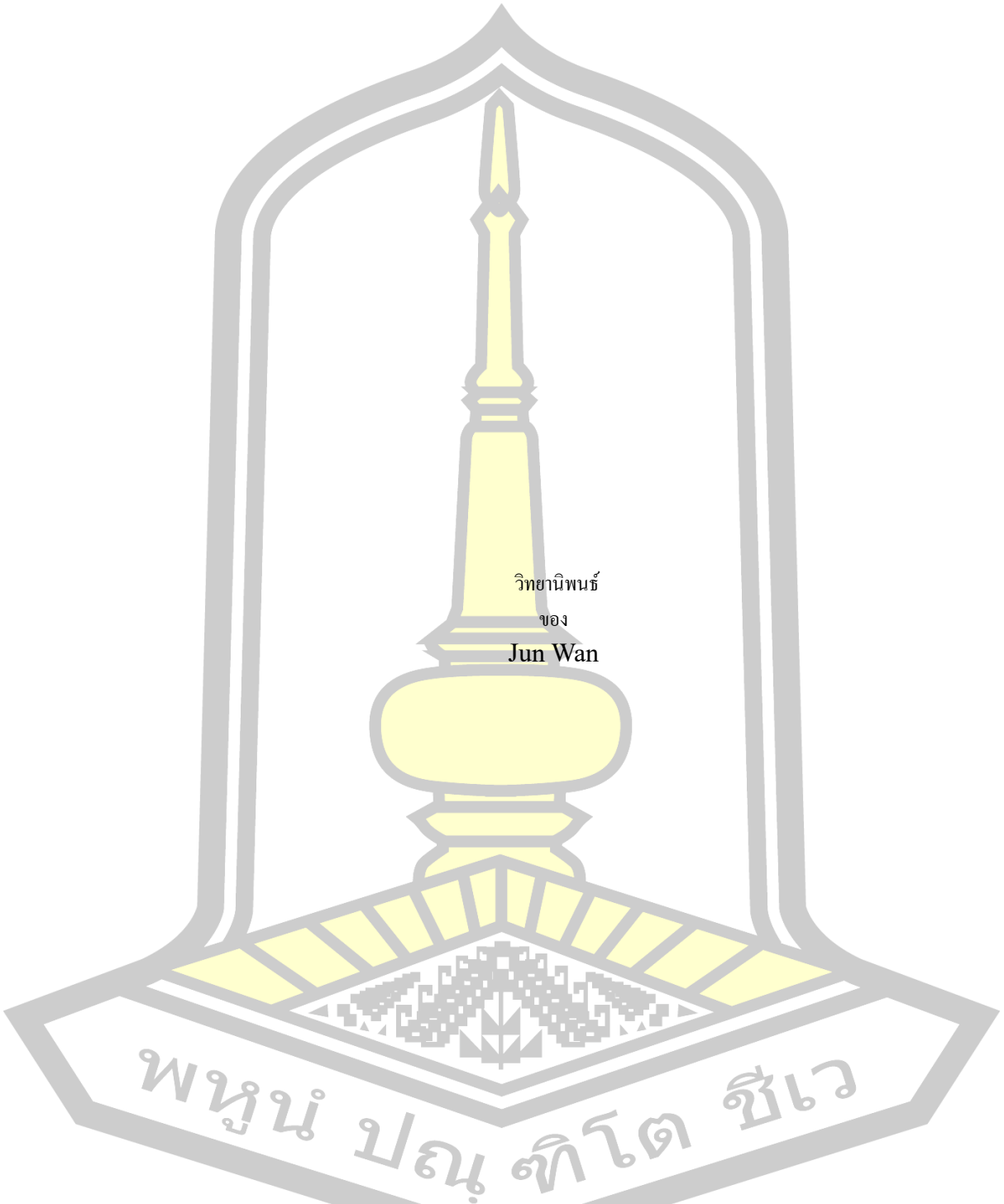
# A Method of Identifying Sentiment from Consumer Multimodality Reviews

Jun Wan

A Thesis Submitted in Partial Fulfillment of Requirements for
degree of Doctor of Philosophy in Computer Science
March 2025

# A Method of Identifying Sentiment from Consumer Multimodality Reviews



วิทยานิพนธ์

ของ

Jun Wan

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาปรัชญาดุษฎีบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

มีนาคม 2568

# A Method of Identifying Sentiment from Consumer Multimodality Reviews

Jun Wan

A Thesis Submitted in Partial Fulfillment of Requirements

for Doctor of Philosophy (Computer Science)

March 2025

The examining committee has unanimously approved this Thesis, submitted by Mr. Jun Wan , as a partial fulfillment of the requirements for the Doctor of Philosophy Computer Science at Mahasarakham University

Examining Committee

.....Chairman
(Asst. Prof. Manasawee Kaenampornpan , Ph.D.)

.....Advisor
(Assoc. Prof. Jantima Polpinij , Ph.D.)

.....Co-advisor
(Assoc. Prof. Gamgarn Sompasertsri , Ph.D.)

.....Committee
(Assoc. Prof. Panida Songram , Ph.D.)

.....External Committee
(Asst. Prof. Bancha Luaphol , Ph.D.)

.....External Committee
(Asst. Prof. Manasawee Kaenampornpan , Ph.D.)

Mahasarakham University has granted approval to accept this Thesis as a partial fulfillment of the requirements for the Doctor of Philosophy Computer Science

.....
(Assoc. Prof. Jantima Polpinij , Ph.D.)
Dean of The Faculty of Informatics

.....
(Prof. Anongrit Kangrang , Ph.D.)
Acting Dean of Graduate School

**TITLE** A Method of Identifying Sentiment from Consumer Multimodality Reviews

**AUTHOR** Jun Wan

**ADVISORS** Associate Professor Jantima Polpinij , Ph.D.
Associate Professor Gamgarn Sompasertsri , Ph.D.

**DEGREE** Doctor of Philosophy **MAJOR** Computer Science

**UNIVERSITY** Mahasarakham University **YEAR** 2025

## ABSTRACT

The rapid expansion of e-commerce and digital media has led to an increasing reliance on online consumer reviews, which play a crucial role in shaping purchasing decisions. Traditional sentiment analysis methods often focus solely on textual data, overlooking the multimodal nature of consumer reviews, which frequently include emoticons and other non-textual elements. This study addresses the challenge of sentiment classification in multimodal consumer reviews by integrating text and emoticons to improve classification accuracy. The primary objective of this research is to develop a sentiment classification method capable of identifying sentiment from movie reviews containing both text and emoticons. Two classification tasks were considered: binary sentiment classification (positive and negative) and multiclass sentiment classification (positive, neutral, and negative). The dataset was collected from the Douban Film and Television Network, comprising Chinese-language movie reviews with embedded emoticons. Textual data was represented using five embedding techniques—Word2Vec, GloVe, FastText, Ada-002, and BERT—while emoticons were encoded using one-hot encoding. The fusion of textual and emoticon features was performed using concatenation. To evaluate model performance, machine learning classifiers such as Support Vector Machine (SVM) and Random Forest, as well as deep learning models like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM), were trained using 10-fold cross-validation. Experimental results demonstrated that deep learning models, particularly LSTM and CNN, outperformed traditional classifiers when combined with contextual embeddings such as BERT and Ada-002. In binary classification, CNN with Ada-002 achieved the highest accuracy, while LSTM with BERT exhibited superior performance in multiclass classification. The inclusion of emoticons enhanced classification results, particularly in deep learning models. For example, in experiments using the first dataset, the Word2Vec + CNN model achieved an accuracy of 0.80 with text alone, which increased to 0.83 when emoticons were included. Similarly, the GLOVE + CNN model improved from 0.80 to 0.83 with the addition of emoticons. Furthermore, two datasets were used to validate the model's performance under different conditions, testing whether the developed model can generalize and maintain good performance when encountering different data.This study highlights the importance of multimodal fusion in sentiment analysis,

demonstrating that integrating emoticons with advanced text representations significantly improves sentiment classification accuracy. These findings provide valuable insights for enhancing sentiment analysis techniques in consumer review analysis.

Keyword : Multimodal Sentiment Analysis, Sentiment Classification, Consumer Reviews, Text and Emoticon Fusion, Machine Learning, Deep Learning, BERT

# ACKNOWLEDGEMENTS

As I approach the completion of this doctoral thesis, my heart is filled with deep gratitude and appreciation. Many individuals have contributed to this journey, and I am truly thankful for their support and guidance.

First and foremost, I extend my sincerest gratitude to my supervisor. Throughout my research on identifying emotions in multimodal consumer reviews, his profound academic insight and unwavering support helped me navigate the complexities of model construction and data processing. His high standards, meticulous guidance, and invaluable advice shaped every stage of my research, from methodology selection to experimental analysis. Moreover, his dedication to rigorous scholarship and relentless pursuit of knowledge have been a constant source of inspiration, motivating me to persevere in my scientific endeavors.

I am also deeply grateful to my classmates and lab partners. During the long hours spent optimizing the emotion recognition algorithm, we exchanged ideas, shared findings, and engaged in insightful discussions on multimodal data fusion and feature extraction. These collaborations often sparked innovation and provided fresh perspectives that enriched my research.

My heartfelt appreciation goes to my family, whose unwavering support has been my greatest source of strength. While I immersed myself in vast datasets and technical challenges, they shouldered daily responsibilities with patience and understanding, allowing me to focus on my work without distraction. Their encouragement has been invaluable throughout this journey.

To all who have contributed in ways both big and small, I extend my deepest gratitude. Your wisdom and support have been instrumental in shaping this research.

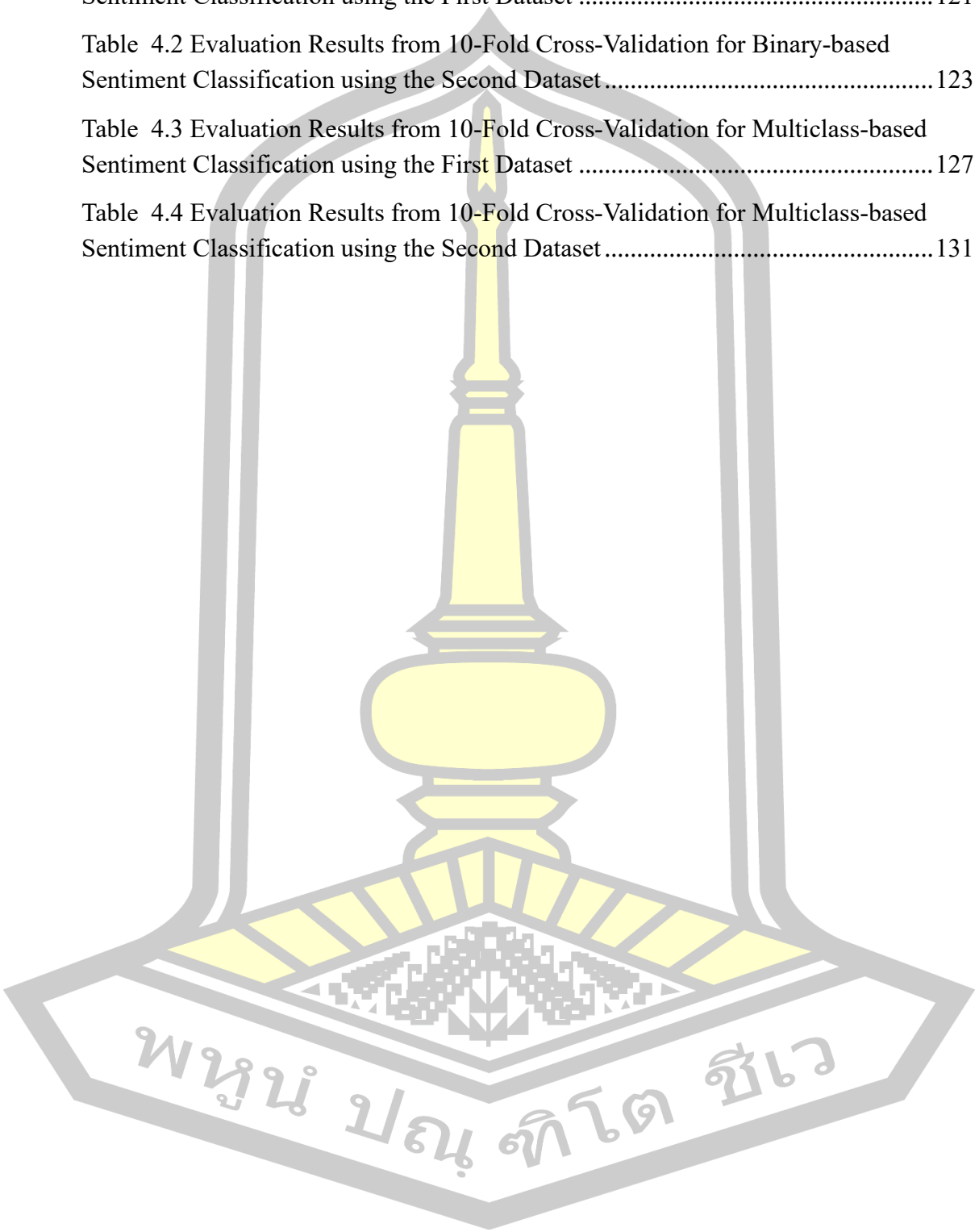
Jun Wan

# TABLE OF CONTENTS

**Page**

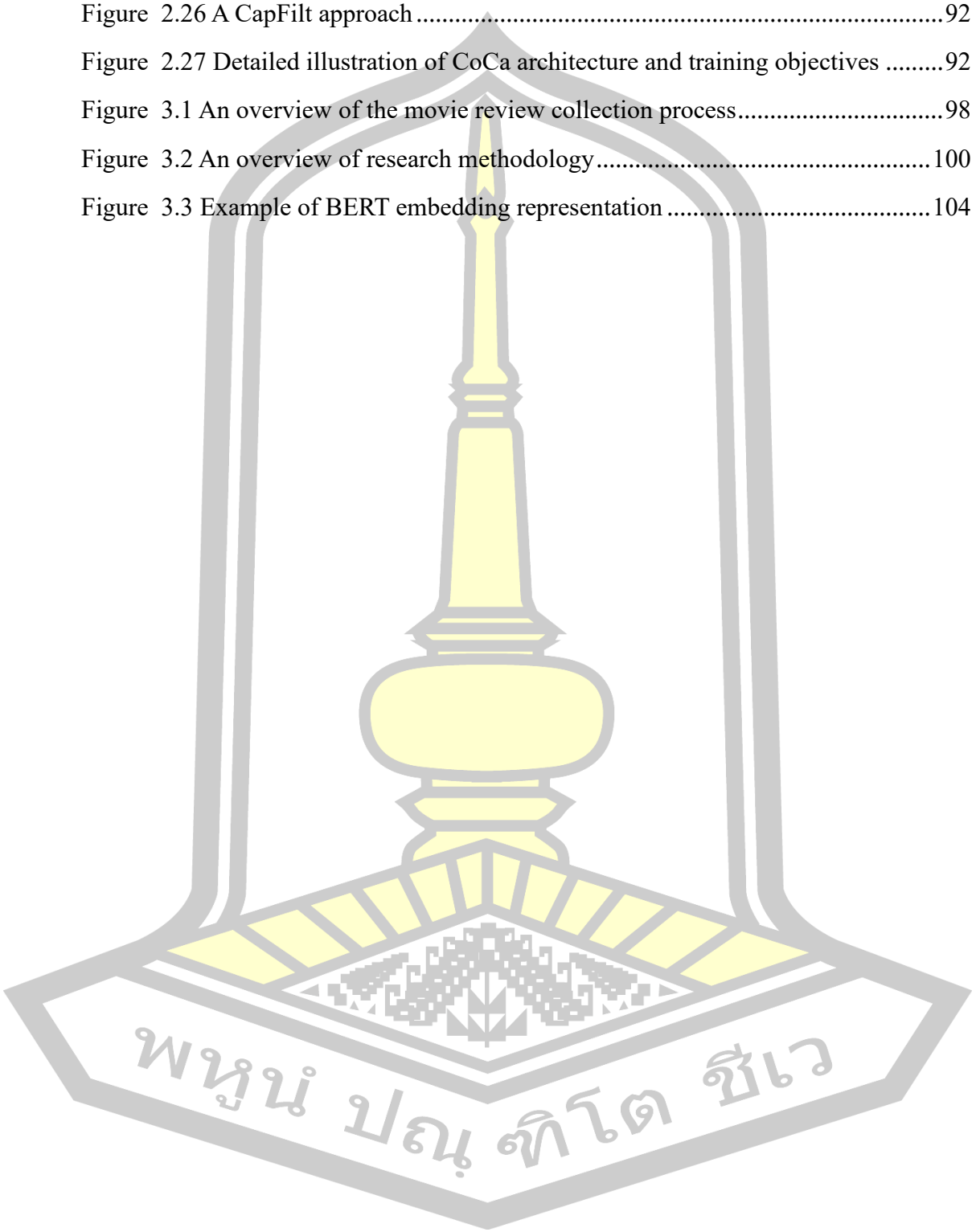
# LIST OF TABLES

**Page**



# LIST OF FIGURES

**Page**



# CHAPTER 1 INTRODUCTION

## 1.1 Background

Due to the rapid growth of intelligent devices, e-commerce, and emerging media technologies, there is an increased focus from consumers, internet and information organizations, and service providers on online comments. Each and every day, hundreds of millions of Internet users engage with the Internet in the online realm. Various activities such as word processing, social communication, internet purchasing, and financial management result in the creation of large amounts of personal data on a daily basis. As an illustration: The 52nd Statistical Report on China's Internet Development, released by the China Internet Network Information Center (CNNIC) in Beijing on August 28, 2023 [1], reveals that as of June 2023, the number of Chinese Internet users has reached 1.067 billion, with an Internet penetration rate of 75.6%. Figure 1.1 illustrates this, highlighting that Chinese Internet users constitute the largest and most engaged group globally. The netizen has transitioned from being a passive recipient of information in the online realm to actively generating and disseminating information in the digital landscape.

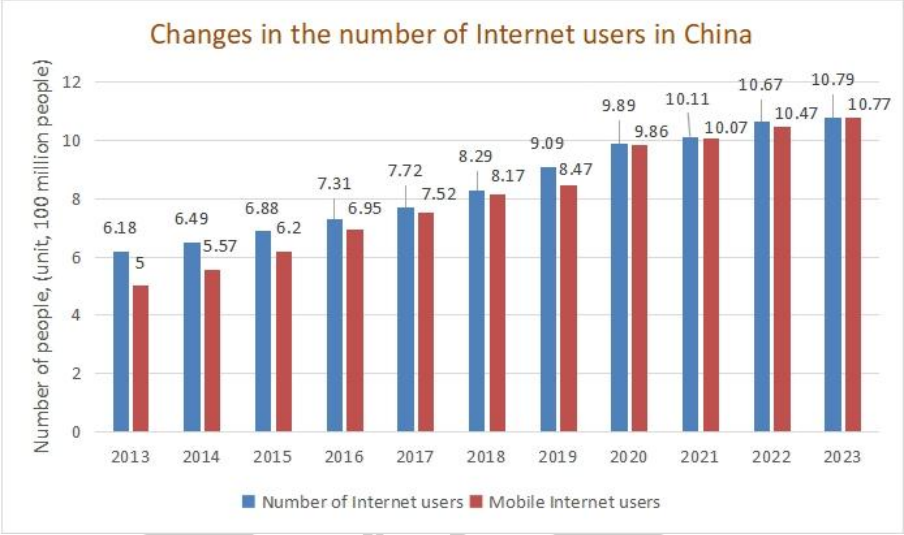



Figure 1.1 Changes in the number of Chinese netizens in recent years

From: https://www.cnnic.com.cn/IDR/ReportDownloads/ 202212/P020221209344717199824.pdf

The multimedia data that netizens publish has expanded beyond the confines of individual text messages and now comprises text, image, and speech. The future purchasing decisions of prospective consumers will be impacted by the sentiments conveyed by Internet users in comment sections or on social media regarding their experiences with the product. Consumers are aided in determining whether an item provides the greatest value by social evaluations. Despite the fact that merchants furnish service descriptions via websites or online e-commerce platforms, user evaluations hold greater trustworthiness, credibility, and persuasiveness, thereby captivating a substantial consumer base. As a result, online evaluations have emerged as a highly regarded source of information among consumers, and the insights ostensibly contained within them significantly influence the purchasing choices of consumers and the revenue of businesses. Nevertheless, as a result of the immense quantity and precision of online evaluations, it can be difficult to glean valuable information from them.

In consumer business, consumer reviews are incredibly valuable in the consumer business, and indeed in many other industries as well. This is because consumer reviews provide direct feedback from guests about their experiences. This feedback can highlight areas where the consumer is excelling and areas where improvements are needed. Positive reviews can build trust with potential consumers and enhance the consumer's reputation. When prospective guests see that others have had positive experiences, they are more likely to choose that business for their stay. Businesses with consistently positive reviews may gain a competitive advantage over others in the area. Consumers are more likely to choose a business with better reviews over one with mediocre or negative reviews. Meanwhile, negative reviews can provide valuable insights into areas where the business may be falling short. By addressing these issues, businesses can improve their services and facilities, ultimately leading to higher consumer satisfaction and loyalty. In addition, Consumer reviews serve as a form of word-of-mouth marketing. Positive reviews can attract new consumers, while negative reviews can provide insights for improvement and demonstrate the business's commitment to consumer satisfaction. In summary, consumer reviews create a feedback loop where businesses can continuously monitor and improve their services based on consumer feedback. This ongoing process of improvement is essential for staying competitive in the hospitality industry.

An approach utilized for analyzing the sentiment or emotional tone expressed in a text is sentiment analysis, which is alternatively referred to as opinion mining [2-4]. This natural language processing (NLP) methodology is applied to the reviews in order to identify the sentiment of the consumers in a piece of text. The goal of sentiment analysis is to classify the sentiment of a text as positive, negative, or neutral.

Nevertheless, consumer evaluations might do indeed comprise multimodal data [5, 6]. Multimodal data encompasses a wide range of formats and modalities, including but not limited to emoticons, textual reviews, images, audio, and video. Certainly, the incorporation of various modalities—such as emoticons, textual reviews, images, audio, and video—in sentiment analysis introduces a number of complexities—yet also provides opportunities for more complex analysis. This is due to the complexity involved in integrating data from various modalities in order to deduce a comprehensive sentiment. Each modality may convey various aspects of sentiment, and it is difficult to determine their relative significance and how to effectively combine them. In addition, specialized techniques are required to extract features from non-textual modalities such as emoticons, audio, video, and images. Illustratively, features in images may consist of colors, objects, or visual components that communicate emotion. Features of an audio file may consist of pitch, tone of voice, or speech patterns. Multimodal data may contain a substantial amount of information and possess high dimensions, thereby presenting difficulties in terms of processing and analysis. Efficiently managing extensive multimodal datasets necessitates the implementation of advanced computational methodologies and infrastructure [7, 8]. Additionally, a semantic disparity may exist between modalities, resulting in differential expressions of the same sentiment. An instance of a direct correspondence between a positive sentiment conveyed in a textual review and positive attributes present in an accompanying image or audio recording may not always occur. Understanding the context in which each modality is conveyed is therefore essential for sentiment analysis to be accurate. The interpretation of sentiment in another modality may be influenced by contextual signals from one modality; therefore, models must capture and exploit these interdependencies. Lastly, it can be difficult and costly to acquire labeled data for training multimodal sentiment analysis models [7-9]. Annotating multimodal data manually frequently necessitates knowledge of numerous domains and modalities.

As mentioned above, sentiment analysis on multimodal data of consumer reviews become a challenge in this study, where it aims to identify consumer's sentiment from consumer reviews containing many data types such as text and emoticons via text classification method in order to classify consumer reviews into positive, neutral, and negative classes. We may need domain experts in language to make a ground truth dataset by determining the labels of each modality found in consumer reviews because each modality may convey various aspects of sentiment, and it is difficult to determine their relative significance and how to effectively combine them.

As mentioned above, this study focuses on the challenge of performing sentiment analysis on multimodal data of consumer reviews. The goal is to identify the sentiment of consumers from consumer reviews that contain various data types, including text and emoticon [10]. This will be achieved through the use of a text classification method, which will classify the consumer reviews into positive and negative categories. In order to create a ground-truth dataset, it may be necessary to involve language domain specialists who can accurately assign labels to each aspect of sentiment found in consumer reviews. This is because different aspects of sentiment may be conveyed through numerous modalities, and it can be challenging to establish their relative importance and how to effectively combine them.

Also, this study presents a sentiment classification model that combines images and text using decision diversity. The text information is considered the primary content, while the image information is utilized to aid in identifying sentiment-related terms within the text. This approach enables the merging of cross-modal features at the feature level. Subsequently, a decision fusion mechanism is devised to combine the decision-level information from the individual modality and the fusion representation, drawing upon the concept of ensemble learning. Ultimately, the decision similarity constraint was incorporated to enhance the variety and comprehensiveness of the model's overall decision.

## 1.2 Research Contribution

Text classification techniques can indeed be applied to identify consumers' sentiment from consumer reviews that include multimodal data such as text and emoticons (or images).

## 1.3 Research Objective

The objective of this study is to present a sentiment classification method able to identifying consumers' sentiment from movie reviews that contain multimodal data, including both text and emoticon (or image). We consider categorizing movie reviews into positive and negative classes.

## 1.4 Research Significances

Studying the application of sentiment classification methods to identify consumers' sentiment from consumer reviews that include multimodality data (text and emoticons (or images)) has numerous significant implications and advantages:

*Comprehensive Understanding of Consumer Sentiment*: Combining textual and visual information enables a comprehensive comprehension of consumer sentiment. This thorough research allows businesses to capture sensitive aspects of relationships with consumers that may not be properly communicated through text alone.

*Enhanced Accuracy and Insightfulness*: By leveraging multiple modalities, sentiment classification models can provide more accurate and insightful sentiment predictions. The inclusion of visual information from images enriches the analysis, leading to more robust sentiment classification outcomes.

*Improved Consumer Experience Management*: Identifying sentiment from multimodality consumer reviews enables businesses to better understand consumer preferences, satisfaction levels, and pain points. This knowledge can inform strategic decisions aimed at enhancing the overall consumer experience and addressing areas of concern.

*Tailored Service Offerings*: Insights gained from multimodality sentiment analysis can guide businesses in tailoring their service offerings to align with consumer expectations and preferences. By identifying recurring themes and sentiments across reviews, businesses can prioritize improvements and optimize their offerings to better meet consumer needs.

*Competitive Advantage*: Utilizing advanced sentiment classification methods for multimodality data analysis can provide businesses with a competitive advantage. By gaining deeper insights into consumer sentiment, businesses can differentiate themselves by delivering superior experiences that resonate with guests and foster loyalty.

*Data-Driven Decision Making*: Multimodality sentiment analysis empowers businesses to make data-driven decisions based on consumer feedback. By systematically analyzing reviews and extracting sentiment trends, businesses can identify actionable insights and prioritize initiatives that have the greatest potential to drive positive outcomes.

*Brand Reputation Management*: Understanding sentiment from consumer reviews helps in managing brand reputation effectively. By promptly addressing negative sentiment and leveraging positive sentiment to enhance brand visibility, businesses can cultivate a positive online reputation and attract more guests.

In summary, applying sentiment classification methods to analyze consumer reviews with multimodality data provides several advantages, such as enhanced comprehension of consumer sentiment, increased service offerings, a competitive advantage, and progress in research and technology. By adopting this strategy, businesses may more effectively cater to the changing requirements and desires of their consumers, ultimately leading to increased satisfaction and loyalty.

## 1.5 Research Scope

1. A sentiment classification method used to identify consumers' sentiment from consumer reviews that contain multimodality data, including both text and images is proposed.
2. This study aims to explore sentiment classification using multimodal data through two distinct approaches:

    (1) Binary-based sentiment classification – This method categorizes sentiments into two classes: positive and negative, providing a straightforward distinction between favorable and unfavorable opinions.

(2) Multiclass-based sentiment classification – In this approach, sentiments are classified into three categories: positive, neutral, and negative, allowing for a more nuanced understanding of varying emotional tones within the data.

By employing both classification methods, this study seeks to evaluate and compare their effectiveness in capturing sentiment patterns in multimodal data.

3. The datasets utilized in this study were collected from the *Douban Film and Television Network website* (https://movie.douban.com/). They consist of Chinese-language movie reviews that incorporate both text and emoticons.

4. Once features are extracted from both texts and images (or emoticons) multimodalities, they can be combined using a data fusion technique. The data fusion method used in this study may be concatenation. It helps combining of features from multiple modalities in a unified representation.

5. Evaluating the performances of sentiment classification models for consumer reviews with multimodality data use many metrices such as accuracy, F1, and the Area Under the ROC Curve (AUC).

## 1.6 Terminologies

1. Multimodal data refers to information that combines multiple types of data or input modalities, such as text, images, audio, and video. In the context of sentiment analysis and natural language processing (NLP), multimodal data enhances understanding by integrating different forms of expression. A common example of multimodal data is text with emoticons, where written language is supplemented with visual symbols to convey emotions more effectively. Emoticons add context, tone, and sentiment nuances that plain text alone may not fully capture, making them an essential component of multimodal sentiment analysis.

2. Consumer reviews with multimodality data refer to feedback or opinions provided by guests about their experiences at a consumer, which include multiple types of information or multimodalities. In this context,

multimodal data typically refers to a combination of textual reviews and accompanying images.

3. Movie reviews are evaluations or critiques of films written by critics or general audiences. They typically analyze various aspects of a movie, such as its plot, acting performances, cinematography, direction, and overall impact. Reviews can be subjective, reflecting personal opinions and emotions, or objective, focusing on technical and artistic elements. They often help audiences decide whether a movie is worth watching and contribute to discussions about its strengths and weaknesses.

4. Sentiment classification model for consumer reviews with multimodality data is to predict the sentiment expressed in reviews using various types of information, including text, images

5. Binary-based sentiment classification is a method of categorizing text into two distinct sentiment categories: positive and negative. It simplifies sentiment analysis by assigning each piece of text to one of these two polarities, making it a widely used approach in natural language processing (NLP) and machine learning. Despite its simplicity, binary sentiment classification may overlook nuanced emotions that fall between positive and negative, which is why more advanced methods, such as multiclass sentiment classification, are sometimes preferred for a more detailed analysis.

6. Multiclass-based sentiment classification is a technique used to categorize text into multiple sentiment classes, typically positive, neutral, and negative. Unlike binary sentiment classification, which only distinguishes between positive and negative sentiments, this approach provides a more nuanced understanding of emotions expressed in text. Multiclass sentiment classification enhances the depth of sentiment analysis, making it valuable for tasks that require a more detailed and comprehensive understanding of user emotions.

# CHAPTER 2 LITERATURE REVIEW

This chapter provides an in-depth review of the key methods, professional terminology, techniques, and fundamental concepts that comprise the sentiment analysis process. Additionally, it draws upon the research literature that serves as a crucial point of reference and a source of inspiration for the implementation of this study.

## 2.1 Sentiment Analysis

### 2.1.1 Definition

Sentiment analysis [11, 12], also known as opinion mining, is a natural language processing (NLP) technique used to determine the sentiment or emotional tone expressed in a piece of text. The goal of sentiment analysis is to classify the sentiment of a given text as positive, negative, or neutral. It has numerous applications across various industries, including market research, consumer feedback analysis, social media monitoring, brand reputation management, and consumer service optimization. It enables businesses to gain insights into public opinion, track sentiment trends over time, and make data-driven decisions based on the emotional tone of textual data. Sentiment classification is indeed one of the tasks within the broader field of sentiment analysis. Sentiment analysis encompasses a range of tasks aimed at understanding and interpreting the sentiment or emotional content present in text data. Sentiment classification, also known as sentiment labeling or sentiment categorization, specifically focuses on the task of classifying text into predefined sentiment categories, such as positive, negative, or neutral.

### 2.1.2 A Generic Framework of Sentiment Classification using Machine Learning

A generic framework for sentiment classification includes many essential processes that can be applied to various domains and types of textual data. A generic framework of sentiment classification can be shown as Figure 2.1 and each processing step is briefed as follows.

***Text Pre-processing*:** The purpose of this step is to eliminate any unnecessary components, such as HTML tags, punctuation, special characters, and useless information. This may also entail activities like as converting to lowercase,

tokenizing, and removing stop words. Additionally, it involves the process of data normalization by applying techniques such as stemming or lemmatization to reduce inflectional forms to their base or root form.

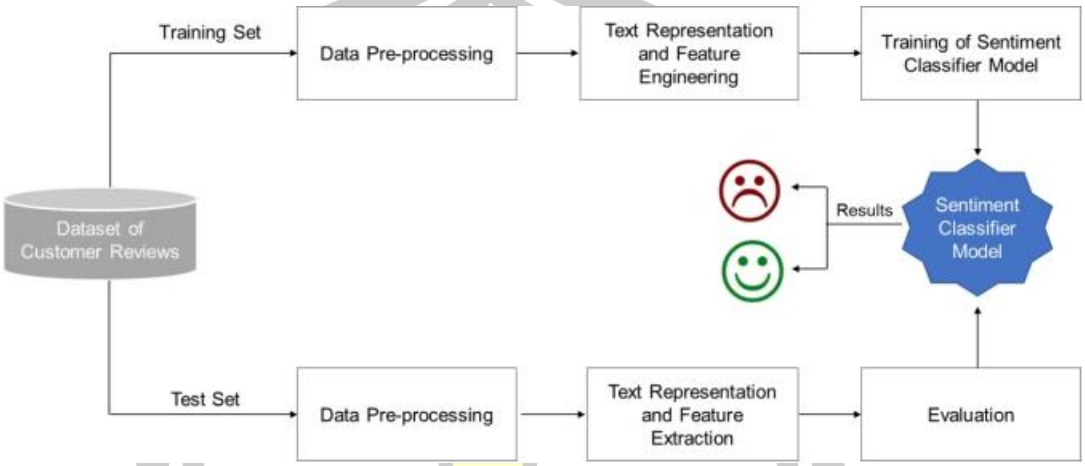



Figure 2.1 A generic framework of sentiment classification

Text Representation and Feature Engineering: Text representation and feature engineering play a crucial role in sentiment classification, as they involve transforming raw text data into a format suitable for machine learning algorithms to process. Here are some common methods for text representation and feature engineering in sentiment classification:

1)Bag-of-Words (BoW) - BOW represents each document as a vector of word counts or frequencies. Words are treated as independent features, and the order of words is ignored. Stop-words and low-frequency words may be removed to reduce noise. BOW is simple and effective but does not capture word semantics or context.

2)Term Frequency-Inverse Document Frequency (TF-IDF) - TF-IDF represents each word in a document by its frequency relative to the entire corpus. Words that are common in a document but rare in the corpus are weighted higher. TF-IDF balances word importance based on frequency and rarity across documents.

3)Word Embeddings - Word embeddings represent words as dense, low-dimensional vectors in a continuous vector space. Word2Vec, GloVe, and FastText are popular pre-trained word embedding models. Word embeddings capture semantic relationships between words and can improve model performance.

4)N-grams - N-grams represent sequences of N contiguous words in a document. Unigrams (single words), bigrams (pairs of adjacent words), and trigrams

(triplets of adjacent words) are common choices. N-grams capture local word dependencies and can provide additional contextual information.

5)Lexicon-based Features - Lexicon-based features involve using sentiment lexicons or dictionaries to extract sentiment-related features from text. Sentiment lexicons contain lists of words annotated with sentiment polarity (e.g., positive, negative, neutral). Features may include sentiment scores, counts of positive and negative words, or polarity ratios.

In the paper, TF-IDF and Word2Vec are used independently, and no direct combination is performed.

TF-IDF: Used for feature selection and document representation, generating statistical features based on term frequency.

Word2Vec: Used to generate word embeddings, capturing semantic information.

If the study requires combining TF-IDF and Word2Vec, a common approach is to use Weighted Word2Vec, where Word2Vec vectors are weighted by their corresponding TF-IDF values. The formula for this approach is as follows:

$$Weighted\_Word2Vec = \sum_{i=1}^{n} TF-IDF(w_i) \cdot Word2Vec(w_i) \quad (2.1)$$

This approach combines the statistical information from TF-IDF with the semantic information from Word2Vec, resulting in more representative text features.

Training Sentiment Classifier Model: This step involves selecting an appropriate machine learning or deep learning algorithm for sentiment classification. Common algorithms include Support Vector Machines (SVM), Naive Bayes, Logistic Regression, Random Forest, CNNs, RNNs, and transformer-based models such as BERT. Many factors influence the selection of an appropriate algorithm for sentiment classification, including the amount and complexity of your dataset, available computational resources, and the desired balance of model performance and interpretability. To develop a sentiment classifier model, it first initializes the chosen algorithm with appropriate hyperparameters, then trains the sentiment classifier model with training set and validates its performance with validation data. If the model produces unsatisfactory results, hyperparameters must be adjusted as appropriate to improve performance and prevent overfitting.

Evaluation of Sentiment Classifier Model: This step is to evaluate the trained model's performance using the test set by using evaluation metrics such as accuracy, precision, recall, F1-score, and confusion matrix to assess its effectiveness in predicting sentiment.

### 2.1.3 A Generic Framework of Sentiment Classification using Deep Learning

A common framework for classification of sentiment using deep learning often consists of multiple essential components and processes. Here is a comprehensive summary of such a framework:

Data Collection and Pre-processing: The purpose of this stage is to gather a dataset consisting of text documents that have been labeled with sentiment classes such as positive, negative, or neutral. Subsequently, the text data is processed by tokenizing it into individual words or sub-words, removing stop-words and punctuation, and using other necessary techniques for text normalization.

Text Representation and Feature Engineering: The purpose of this stage is to transform the text data into a format that is appropriate for feeding into a deep learning model. Typical methods include using word embeddings such as Word2Vec and GloVe, character embeddings, or sub-word embeddings like Byte Pair Encoding. Additionally, it involves the implementation of feature engineering techniques to extract supplementary features from the text data, such as n-grams, part-of-speech tags, or syntactic features.

Model Architecture Design: The purpose of this stage is to design the structure of the deep learning model specifically for sentiment categorization. Typical architectures comprise Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), Gated Recurrent Units (GRUs), or Transformer-based models such as BERT. Additionally, it selects the suitable layers, activation functions, and regularization approaches according to the specific attributes of the data and the objective at hand.

Model Training: The purpose of this stage is to split the dataset into training, validation, and test sets. The training set is utilized for model training, the validation set is employed for hyperparameter tuning and performance monitoring, and the test set is utilized for final model evaluation. Subsequently, the dataset will undergo training with a deep learning model utilizing techniques such as stochastic gradient

descent (SGD), Adam optimizer, or other optimization algorithms. Subsequently, it is necessary to monitor the training process, which entails monitoring the convergence of loss, the duration of training, and the performance metrics on the validation set.

Hyperparameter Tuning: This step involves doing hyperparameter adjusting to enhance the performance of the model. It focuses on optimizing hyperparameters such as learning rate, batch size, dropout rate, number of layers, and hidden units. Subsequently, it employs methodologies such as grid search, random search, or Bayesian optimization to effectively explore the hyperparameter space.

Model Evaluation: The purpose of this stage is to assess the performance of the trained model on the test set, which consists of unseen data. Evaluation metrics, including as accuracy, precision, recall, and F1 score, are then calculated based on the task at hand. Additionally, it examines the model's predictions and errors in order to have a deeper understanding of its capabilities and limitations.

## 2.2 Sentiment Analysis on Multimodality Data

Sentiment classification on multimodal data refers to the task of predicting sentiment labels (such as positive, negative, or neutral) from a combination of multiple types of data modalities, typically including both textual and non-textual information (e.g., emoticons, images) [11, 12]. In the context of multimodal sentiment classification for consumer reviews, the modalities typically considered are text, emoticons, and images.

### 2.2.1 Examples of Multimodal Data

***Text-Image Data*:** A common form of multimodality data is combining textual information with image representations [13]. Within the context of social media, it is customary for users to create posts that comprise of a written description, also referred to as a caption, along with a corresponding image or video. The incorporation of written and visual components in analysis can result in deeper insights, as demonstrated in Figure 2.2.

***Text-Audio Data*:** Textual transcriptions of spoken information are frequently employed in conjunction with audio data in applications such as speech recognition and sentiment analysis in order to improve accuracy and understanding.

***Text-Video Data*:** Video analysis is the integration of subtitles or transcriptions of spoken words with the visual data. This combination enables the execution of tasks such as video summarization, object recognition, and sentiment analysis.


Figure 2.2 Multimodality image-text data examples

***Text-Emoticon Data*:** This data pertains to emoticons or emojis that are embedded inside textual information and are represented as text-emoticon data. Emoticons are textual combinations inputted on a keyboard that serve the purpose of conveying emotions, moods, or sentiments in written communication. Emojis, on the other hand, are graphical representations of face expressions, objects, and symbols. They are frequently used in written communication to express similar feelings and convey similar thoughts. Text-emoticons are frequently used in various forms of digital communication, such as text messages, emails, social media posts, and online reviews. They augment the manifestation of tone and ardor by integrating emotional context into written language. Examining data on text-emoticons can provide significant insights for activities such as sentiment analysis, emotion recognition, and evaluating the emotional expression of text-based content. Researchers and data scientists frequently utilize text-emoticon analysis to acquire insights into the emotional content of textual data. Emoticons have become indispensable in modern internet communication, playing a vital role in bridging the gap between written text and nonverbal signals. They greatly improve the ability to convey emotions and meaning in digital messaging. Here are numerous often employed emoticons that symbolize the emotions of others.

-:) or :-) - Represents a smiling face.

-:( or :-( - Represents a sad face.

-:D or :-D - Represents a wide smile or laughter.

-<3 - Represents a heart, often used to express love or affection.

-:/ - Represents uncertainty or skepticism.

-:P or :-P - Represents sticking out one's tongue playfully.

## 2.3 Consumer Reviews with Multimodality Data

### 2.3.1 Definition

Consumer reviews with multimodality data are feedback or opinions made by guests regarding their consumer experiences that incorporate a variety of information or multimodalities [14]. In this context, multimodal data is often defined as a combination of textual reviews and accompanying images.

***Text*:** Guests can provide comprehensive reviews that highlight specific aspects of their stay, such how well-kept the rooms were, how well the service was, or what facilities were offered.

***Images*:** Guests can submit images of their consumer rooms, the facilities that the consumer offers, or any other relevant images that can present a visual representation of their experience. This can be especially useful for displaying the atmosphere, furnishings, or any problems that may have arisen.

***Videos*:** Guests may record videos of their consumer stay, including the check-in process, the view from their room, and their overall impression of the consumer. Videos provide a more immersive and dynamic representation of the experience.

***Audio*:** Guests can have the option to provide audio recordings expressing their opinions and experiences on the consumer. This mode of communication can be useful for capturing more nuanced aspects such as tone of voice or emotional responses.

***Emoticons*:** Emoticons, or emojis, are visual depictions of facial expressions or symbols that are employed to portray emotions, reactions, or feelings in digital communication. Within the realm of consumer evaluations, guests often employ emoticons to concisely convey their sentiments or enhance the emotional depth of their feedback. Emoticons can be used to express good or negative feelings, levels of

pleasure, or specific emotions related to various components of the consumer experience.

By integrating multimodality into consumer reviews, prospective visitors can access a wide array of information and experiences to make well-informed selections. Consumer managers can derive advantages from this input by acquiring insights into certain areas of enhancement and efficiently addressing guest issues.

### 2.3.2 Examples of Multimodality of Consumer Review

Figure 2.3 shows a few examples of multimodality consumer reviews that incorporate both text and emoticon.

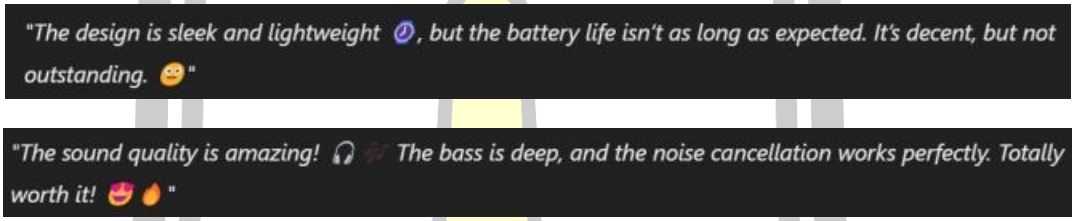



Figure 2.3 Examples of Multimodality Consumer Reviews Incorporating both Text and Emoticon

### 2.3.3 The Key Challenges in Multimodality Sentiment Analysis

The following are the primary challenges in multimodality sentiment analysis at present [15]:

***Feature Representation*:** Feature representation in multimodality sentiment analysis presents several challenges due to the heterogeneous nature of the data, which includes both textual and non-textual (e.g., image, audio, video) modalities. Several significant challenges are as follows:

1) ***Semantic Heterogeneity*:** Textual and visual multimodalities may convey sentiment in different ways, leading to semantic heterogeneity. For example, sentiments expressed in text may not always align with sentiments expressed in images. Integrating these heterogeneous multimodalities while preserving their unique semantic characteristics poses a challenge.

2) ***Multimodality Fusion*:** Combining features from different multimodalities (e.g., text and images) to create a unified representation is non-trivial. Effective multimodality fusion techniques

are needed to capture complementary information from each modality while minimizing redundancy and noise.

3) ***Feature Dimensionality*:** Multimodality datasets often result in high-dimensional feature spaces, especially when dealing with large-scale textual and visual data. Managing the dimensionality of features and selecting informative features that contribute to sentiment analysis while avoiding overfitting is challenging.

4) ***Data Sparsity*:** In multimodality sentiment analysis, textual and visual features may exhibit varying levels of sparsity. Textual data may suffer from sparse representations due to the vast vocabulary and long-tailed distribution of words, while visual data may suffer from sparse feature vectors or missing values.

5) ***Cross-Modal Discrepancy*:** Multimodalities such as text and images may capture different aspects of the same underlying sentiment, leading to cross-multimodality discrepancy. Aligning representations across modalities to ensure consistency and coherence in sentiment analysis is challenging, especially when dealing with diverse and heterogeneous data sources.

6) ***Contextual Understanding*:** Understanding the contextual relationship between different modalities is crucial for accurate sentiment analysis. However, capturing contextual information and dependencies between textual and visual features in a multimodal context remains a challenge, particularly in complex scenarios where context may be ambiguous or implicit.

Addressing these challenges requires the development of innovative feature representation techniques, multimodality fusion strategies, and machine learning algorithms tailored to multimodal sentiment analysis. Researchers continue to explore novel approaches, such as attention mechanisms, cross-multimodality embeddings, and multimodality fusion networks, to overcome these challenges and improve the accuracy and interpretability of multimodality sentiment analysis models.

***Data Fusion*:** Data fusion in multimodality sentiment analysis refers to the process of integrating information from multiple modalities, such as text, images, audio, and video, to analyze and understand the sentiment expressed in a piece of

content. While data fusion offers the potential to leverage complementary information from diverse multimodalities, it also poses several challenges. Here are some of the main challenges:

1) ***Heterogeneity of Multimodalities*:** Different modalities convey information in distinct formats and structures. For example, text is represented as sequences of words, while images consist of pixels. Integrating these heterogeneous multimodalities requires careful consideration of their unique characteristics and representation methods.

2) ***Semantic Alignment*:** Multimodalities may capture different aspects of sentiment, leading to semantic misalignment. For instance, text and images may express sentiment towards different features or aspects of the same entity. Aligning the semantics across modalities to ensure coherent sentiment analysis is challenging.

3) ***Feature Dimensionality*:** Fusion of multiple modalities often results in high-dimensional feature spaces. Managing the dimensionality of fused features is crucial to prevent computational complexity, overfitting, and the curse of dimensionality.

4) ***Modality Weighting and Importance*:** Each modality may contribute differently to sentiment analysis, and their relative importance may vary depending on the context. Determining the optimal weighting of modalities and identifying the most informative features within each modality pose challenges in data fusion.

5) ***Data Sparsity and Missing Modalities*:** Modalities may exhibit varying levels of sparsity, with some modalities having missing or incomplete data. Handling sparse and missing modalities during fusion is essential to ensure robust and reliable sentiment analysis.

6) ***Cross-Modal Correlation*:** Understanding the correlation and dependencies between different modalities is crucial for effective data fusion. Capturing cross-modal correlations and exploiting complementary information while avoiding redundancy and noise is a challenging task.

Addressing these challenges requires the development of innovative fusion techniques, including multimodal feature extraction methods, fusion architectures, and learning algorithms tailored to multimodal sentiment analysis. Researchers continue to explore new approaches, such as attention mechanisms, cross-modal embeddings, graph-based fusion, and deep learning architectures, to overcome these challenges and advance the field of multimodal sentiment analysis.

***Cross-Modality Alignment*:** Cross-modality alignment in multimodal sentiment analysis refers to the process of aligning representations or features extracted from different modalities (e.g., text, images) to enable effective fusion and analysis. While cross-modality alignment offers the potential to leverage complementary information from diverse modalities, it also poses several challenges. Here are some of the main challenges:

1) ***Semantic Misalignment*:** Different modalities may capture different aspects of sentiment or convey sentiment using distinct linguistic or visual cues. Aligning these heterogeneous representations to ensure semantic coherence and consistency in sentiment analysis is challenging.

2) ***Feature Space Discrepancy*:** Modalities often have different feature spaces or dimensionalities, making direct comparison and fusion difficult. Aligning feature spaces across modalities while preserving relevant information and minimizing information loss poses a significant challenge.

3) ***Domain Discrepancy*:** Modalities may originate from different domains or sources, leading to domain-specific characteristics and biases. Aligning representations across domains to ensure generalizability and robustness in sentiment analysis is challenging, especially in cross-domain scenarios.

4) ***Scale and Variability*:** Modalities may vary in scale, granularity, and variability, making alignment challenging. Handling differences in scale and variability between modalities while maintaining alignment and consistency is crucial for effective fusion.

5) ***Modality-Specific Noise*:** Each modality may contain noise or irrelevant information that is specific to that modality. Aligning

representations while filtering out modality-specific noise and preserving informative features poses challenges in cross-modality alignment.

6) ***Data Sparsity and Missing Modalities*:** Modalities may exhibit varying levels of sparsity, with some modalities having missing or incomplete data. Aligning representations while handling sparse and missing modalities requires robust alignment techniques.

7) ***Cross-Modal Correlation*:** Understanding the correlations and dependencies between different modalities is crucial for effective alignment. Capturing cross-modal correlations and exploiting complementary information while avoiding redundancy and noise is a challenging task.

Addressing these challenges requires the development of innovative cross-modality alignment techniques, including unsupervised alignment methods, domain adaptation approaches, and deep learning architectures tailored to multimodal sentiment analysis. Researchers continue to explore new strategies to overcome these challenges and advance the field of cross-modal sentiment analysis.

***Emotion Recognition*:** Emotion recognition in multimodal sentiment analysis involves detecting and understanding emotions expressed in various modalities, such as text, images, audio, and video. While multimodal approaches offer the potential to capture richer emotional cues, they also pose several challenges. Here are some of the main challenges:

1) ***Semantic Heterogeneity*:** Different modalities may convey emotions in different ways, leading to semantic heterogeneity. For example, textual expressions of emotion may differ from facial expressions or vocal intonations. Integrating these heterogeneous modalities while preserving their unique emotional cues poses a challenge.

2) ***Modality Fusion*:** Combining emotional information from different modalities requires effective fusion techniques. Integrating textual, visual, and auditory cues to capture a holistic understanding of emotions while avoiding redundancy and noise is challenging.

3) ***Cross-Modality Alignment*:** Aligning emotional representations across different modalities is crucial for effective fusion. Understanding the

correlations and dependencies between textual, visual, and auditory cues and aligning them to ensure consistency in emotion recognition poses challenges.

4) ***Feature Representation*:** Extracting informative features from multimodal data is challenging. Representing emotional cues from text, images, audio, and video in a unified and meaningful way requires innovative feature representation techniques.

5) ***Emotion Complexity*:** Emotions are complex and multifaceted, often involving combinations of affective states. Recognizing nuanced emotions and capturing subtle variations in emotional expression across modalities is challenging.

6) ***Contextual Understanding*:** Emotions are influenced by context, including situational, cultural, and individual factors. Understanding the contextual cues and dependencies between different modalities is crucial for accurate emotion recognition in multimodal sentiment analysis.

7) ***Labeling and Annotation*:** Collecting labeled data for emotion recognition in multimodal settings can be challenging and resource-intensive. Annotation of emotional cues in diverse modalities requires specialized expertise and may suffer from subjective biases.

Addressing these challenges requires interdisciplinary research combining techniques from natural language processing, computer vision, signal processing, and affective computing. Innovative approaches, including deep learning architectures, attention mechanisms, cross-modal embeddings, and multimodal fusion networks, are being explored to overcome these challenges and advance the field of multimodal emotion recognition and sentiment analysis.

***Co-learning*:** Co-learning in multimodal sentiment analysis refers to the process of jointly learning representations or models from multiple modalities to improve performance in sentiment analysis tasks. While co-learning offers the potential to leverage complementary information from diverse modalities, it also poses several challenges. Here are some of the key challenges:

1) ***Heterogeneity of Modalities*:** Different modalities may have distinct characteristics and structures, making joint learning challenging.

Integrating heterogeneous modalities while preserving their unique features and capturing cross-modal dependencies is a significant challenge.

2) ***Cross-Modal Alignment*:** Aligning representations across modalities to ensure coherence and consistency is crucial for effective co-learning. Understanding the correlations and dependencies between different modalities and aligning them appropriately pose challenges, especially in the presence of semantic heterogeneity.

3) ***Feature Fusion and Integration*:** Integrating features from different modalities while avoiding redundancy and noise is challenging. Developing fusion techniques that effectively combine textual, visual, auditory, and other modalities to enhance sentiment analysis performance requires innovative approaches.

4) ***Data Sparsity and Missing Modalities*:** Modalities may exhibit varying levels of sparsity, with some modalities having missing or incomplete data. Handling sparse and missing modalities during co-learning poses challenges in terms of representation and fusion.

5) ***Semantic Interpretation*:** Learning meaningful representations across modalities requires capturing the semantic relationships between textual, visual, and auditory cues. Developing models that can effectively interpret and represent the semantics of multimodal data poses challenges, especially in complex scenarios.

6) ***Labeling and Annotation*:** Collecting labeled data for multimodal sentiment analysis can be challenging and resource-intensive. Annotation of sentiment in diverse modalities requires specialized expertise and may suffer from subjective biases.

Addressing these challenges requires interdisciplinary research combining techniques from natural language processing, computer vision, audio processing, and machine learning. Innovative approaches, including deep learning architectures, multimodal fusion networks, attention mechanisms, and transfer learning techniques, are being explored to overcome these challenges and advance the field of multimodal sentiment analysis.

### 2.3.4 Real-world Applications of Multimodal Sentiment Analysis

Multimodal sentiment analysis provides a wide range of practical uses in several fields like as e-commerce, social data analysis, healthcare, and content recommendation [16].

***Social Date Monitoring*:** This study examines the emotion of social media posts, which commonly include text, emoticons, images, and videos, to determine public opinion on certain items, events, or themes. For example, social network websites can monitor users' emotional changes in real time and deliver services based on their preferences using text, photo, video, and voice data.

***e-commerce*:** This is an analysis of consumer reviews that include text and, in many cases, images or videos in order to determine consumer satisfaction and areas for improvement. For example, e-commerce websites can use voice and expression data to assess the user's emotional state and propose suitable products.

***Content Recommendation*:** This study aims to improve recommendation systems by taking into consideration consumers' multimodal content preferences.

***Market Monitoring*:** By analyzing the sentiment of multi-media advertisements, this study examines their effectiveness. As an illustration, Spanish academics offered forward a novel tool in 2021—a sentiment index derived from newspapers—to real-time monitor economic activity in Spain. This measure not only outperforms the well-known economic attitudes measure of the European Commission, but it also predicts Spain's GDP exceptionally well [17].

***Healthcare*:** This study examines patient reviews and comments, which may consist of both written text and audio recordings, with the aim of understanding patient attitudes and improving healthcare services. For instance, this method can successfully assess the interaction between patients and doctors as well as their satisfaction with the quality of medical treatments [18]. In the healthcare domain, the diverse forms of online reviews provide usable material for multimodal sentiment analysis.

To the best of our knowledge, deep learning-based multimodal sentiment analysis technology has diverse applications in several real-world settings, including human-computer interaction, smart home systems, intelligent consumer service, and online education. For instance, in the realm of education, students' emotional

fluctuations can be detected over time by scrutinizing their emotional data, such as voice and text. This enables timely adjustments to teaching approaches and enhances students' learning outcomes. In the domain of smart home technology [19], environmental factors such as room temperature and humidity can be automatically regulated by detecting speech emotion, hence enhancing the overall comfort of the room.

## 2.4 Related Theorems, Algorithms, and Concepts

Deep learning methods, including Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU), have become popular for multimodal sentiment analysis in recent years [20, 21].

### 2.4.1 Recurrent Neural Network (RNN)

RNN, or Recurrent Neural Network, is a widely used model for processing sequential data. It has shown significant advancements in various tasks such as dialogue, text comprehension, and machine translation. RNN has become one of the dominant models in natural language processing, particularly for handling sequence data. Unlike conventional feed forward neural networks, RNNs include the ability to incorporate information from past states, allowing them to effectively simulate each piece in a sequence and exhibit a "*memory*" function. The model typically accepts sequence data as input, successfully captures the relationship features between sequences through the internal structural design of the network, and normally produces output in the form of a sequence. Figure 2.4 shows the structure of RNN.

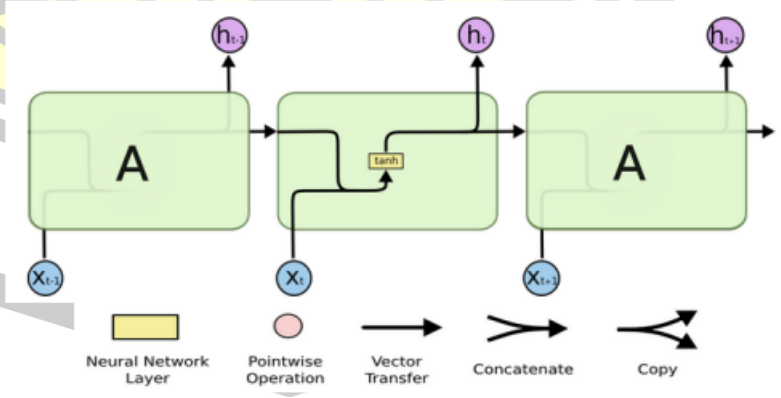



Figure 2.4 A Structure Diagram of RNN

From: https://www.datacamp.com/tutorial/tutorial-for-recurrent-neural-network

The input comprises two components: $h(t-1)$ and $x(t)$, which respectively denote the output of the hidden layer at the previous time step and the input at the current time step. Once they are introduced into the RNN architecture, they will undergo a process of "*fusion*" where they are combined together. The value of $h$ at time $t-1$. Afterwards, this newly formed tensor will be processed by a fully connected layer, also known as a linear layer. This layer utilizes the hyperbolic tangent function (tanh) as its activation function to produce the final output $h(t)$ for the current time step. This output, together with $x(t+1)$ for the subsequent time step, will then be inputted into the system, and the process will continue in this manner. Based on the structure of RNN, the internal calculation formula is derived:

$$h_t = \tanh([W_t, h_{t-1}] + b_t) \tag{2.2}$$

In RNN, it applies tanh as the activation function. The tanh function helps manage the values coming through the network by compressing values between -1 and 1.

Because of its uncomplicated internal structure and little computational resource demands, it possesses significantly less parameters compared to subsequent RNN variations such as LSTM and GRU models. Additionally, it exhibits strong performance on tasks involving short sequences.

However, traditional RNNS have disadvantages as well. The performance of traditional RNN in resolving the association between lengthy sequences has been demonstrated to be inadequate through practice. The reason for this is that excessively lengthy sequences during backpropagation result in erroneous gradient calculations, gradient disappearance, or explosion.

As mentioned above, the gradient calculation can be reduced using the backpropagation method and the chain rule, resulting in the following formula:

$$D_n = \sigma'(z_1)\omega_1 \cdot \sigma'(z_2)\omega_2 \cdot \ldots\ldots \cdot \sigma'(z_n)\omega_n \tag{2.3}$$

The derivative of the sigmoid function has a constant range of values between 0 and 0.25. When the input value, represented by w, is less than 1, the gradient of the function will become extremely tiny. This phenomenon is referred to as gradient vanishing. Conversely, if we deliberately amplify the value of w to a

number greater than 1, the continuous multiplication can result in an excessively huge gradient, which is referred to as gradient explosion.

However, if the gradient vanishes during the training process, the weights become unmodifiable, ultimately resulting in the failure of the training. The gradients resulting from the gradient explosion are excessively enormous to make significant updates to the network parameters, and in severe situations, the outcomes may exceed the maximum value.

In summary, RNN is a type of neural network that possesses intrinsic self-connections. The previous output of a neuron can be used as the input for the same neuron in the next time step. The unfolded recurrent neural network (RNN) may capture the repeating loop structure and share parameters, resulting in a significant reduction in the number of network parameters. The primary purpose of a Recurrent Neural Network (RNN) is to establish a correspondence between an input sequence and an output sequence. Unlike other models, an RNN does not necessitate the output sequence to have the same length as the input sequence, making it suitable for processing variable-length sequence data. However, throughout the trials, the researchers discovered that the original recurrent neural network (RNN) struggles to accurately represent long-range connections. This is due to the fact that as the number of loop iterations increases, the RNN tends to lose and struggle to retain important long-term information. Furthermore, mitigating the issue of vanishing or exploding gradients in RNNs is also a formidable challenge. Consequently, researchers have suggested multiple superior variations to enhance the initial RNN model.

### 2.4.2 Long Short-Term Memory (LSTM)

LSTM is an enhanced model of RNN [22, 23] that addresses the constraints of ordinary RNNs, such as the issue of gradient vanishing or exploding in real-world scenarios. In contrast to the normal RNN, the LSTM maintains a consistent fundamental structure. At each time step, it receives the hidden state from the previous time step and the input from the current time step. The standard LSTM cell structure is seen in Figure 2.5. A comprehensive empirical investigation in the field of literature has been carried out to examine the recurrent network architecture [24]. One of the key findings is that in LSTM networks, the forgetting gate has the highest level of significance, followed by the input gate and the output gate.

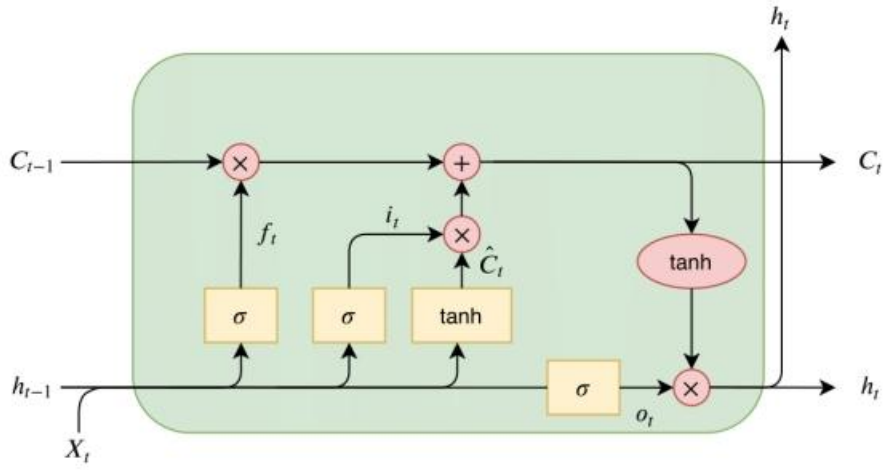



Figure 2.5 A structure diagram of LSTM

From: https://thorirmar.com/post/insight_into_lstm/

LSTM can successfully mitigate the issues of vanishing recurrence and exploding gradients that may arise in lengthy sequence situations. While it cannot completely eradicate this effect, it outperforms regular RNNs when it comes to longer sequence problems. The LSTM architecture consists of three gates, namely the input gate, forget gate, and output gate, as well as the cell state, also known as the cell memory. The forget gate is responsible for selectively discarding information from the cell memory at the previous time step. The forget gate, denoted as $f(t)$, is represented by following formula.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.4}$$

The internal structure calculation of a traditional RNN involves concatenating the input $x(t)$ at the current time step with the hidden state $h(t\text{-}1)$ at the previous time step to obtain $[x(t), h(t\text{-}1)]$. This concatenated input is then passed through a fully connected layer and activated by the sigmoid function to obtain $f(t)$. The function $f(t)$ can be conceptualized as the gate value, similar to the action of opening and closing a door. This gate value influences the tensor that passes through it. Additionally, the forgetting gate value affects the cell state of the subsequent layer, indicating the amount of information that has been disregarded in the past. The calculation of the forgetting gate value is based on $x(t)$ and $h(t\text{-}1)$. The formula determines the extent to which the cell state of the previous layer should forget past knowledge, based on the current time step input and the last time step hidden state $h(t\text{-}1)$.

The sigmoid activation function performs the following formula:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.5}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_c) \tag{2.6}$$

There are two formulas available for computing the input gate. The first formula calculates the value of the input gate, which is similar to the forgetting gate formula, although it is employed for a different purpose. This formula represents the extent to which the input information must be processed or refined. The second formulation of the input gate is equivalent to the calculation of the internal structure in a typical RNN. Unlike the standard RNN, LSTM receives the current cell state instead of the concealed state. The formula of cell state update calculation can be written as follows.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{2.7}$$

The cell update process is straightforward and can be easily comprehended. It does not involve a fully connected layer. Instead, it consists of multiplying the forgotten gate value obtained at the current time step with the previous time step's cell state, denoted as $C(t\text{-}1)$, and adding it to the product of the input gate value and the current time step's non-updated cell state, denoted as $C(t)$. Ultimately, the revised $C(t)$ is acquired as a component of the input for the subsequent time step. The entire process of updating the cell state involves the utilization of both the forget gate and the input gate. Then, the formula of output gate calculation can be provided as follows.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.8}$$

$$h_t = o_t * \tanh(C_t) \tag{2.9}$$

The first formula is to calculate the gate value of the output gate, which is calculated in the same way as the forget gate and input gate. The second formula is to use this gate value to generate the hidden state $h(t)$, which will act on the updated cell state $C(t)$ and do tanh activation, and finally obtain h(t) as part of the input for the next time step. The whole process of output gate is to produce the hidden state $h(t)$.

### 2.4.3 Bi-LSTM

Bi-LSTM refers to a bidirectional LSTM [25]. It maintains the internal structure of the LSTM while applying it twice with different orientations. The results acquired from the two applications are then concatenated to form the final output, as depicted in Figure 2.6.

Figure 2.6 demonstrates that the text "*I love China*." undergoes two separate LSTM processes, one from left to right and one from right to left. The resulting tensors from each process are then combined by concatenation to form the final output. This structure has the ability to capture some distinct characteristics of language grammar before or after a certain event, and improve the connection between meaning and context. However, it also results in the model parameters and computing complexity being doubled. Typically, an assessment of the corpus and computing resources is necessary to determine whether to employ this structure.

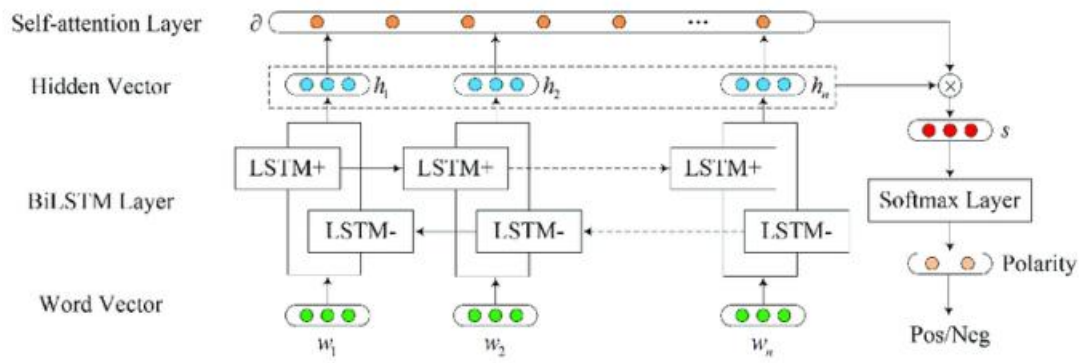



Figure 2.6 The working principle of Bi-LSTM

From: https://www.researchgate.net/figure/The-architecture-of-a-Self-Attention-Based-BiLSTM-neural-network-model_fig5_337748590

### 2.4.4 Gated Recurrent Units (GRU)

GRU, which stands for Gated Recurrent Unit, is a widely used variation of the RNN [26]. A structure of GRU can be shown as Figure 2.7. Unlike LSTM cells, GRU cells consist of only two types of gates: (1) Reset Gate and (2) Update Gate. The Update Gate can be considered as being connected to the input gate and the forget gate in LSTM cells. GRU exhibits comparable performance to LSTM, although due to the absence of one gate, it boasts a decreased parameter count and simpler, more straightforward calculations. Consequently, GRU finds extensive application in actual

investigations. Furthermore, empirical evidence from studies has shown that GRU surpasses LSTM in all tasks except for language modeling.

The update gate, denoted as $z(t)$, and reset gate, denoted as $r(t)$, are determined by concatenating $X(t)$ and $h(t-1)$ for linear transformation, followed by activation using a sigmoid function. Subsequently, the reset gate value is employed to modulate the influence of $h(t-1)$, determining the extent to which information from the preceding time step can be utilized. The previous state $h(t-1)$ is utilized to carry out the fundamental RNN calculation, which is combined with $x(t)$ for linear transformation, and subsequently activated by the hyperbolic tangent function (tanh) to obtain the updated state $h(t)$. Ultimately, the revised gate value will influence the new $h(t)$. The 1-gate value is applied to $h(t-1)$, and the outcomes of both are combined to yield the ultimate hidden state output $h(t)$. This technique entails that the update gate possesses the capacity to preserve the previous outcome. When the gate value approaches 1, the output becomes the updated $h(t)$. Conversely, when the gate value approaches 0, the output becomes the $h(t-1)$ from the previous time step.

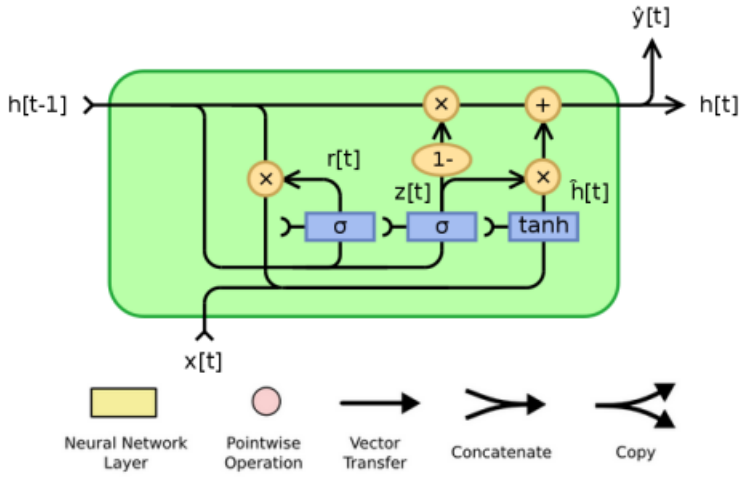



Figure 2.7 A Structure of GRU

From: https://paperswithcode.com/method/gru

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{2.10}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2.11}$$

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t]) \quad (2.12)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \quad (2.13)$$

In summary, GRU and LSTM are two types of RNNs that effectively address the issue of gradient disappearance or explosion when capturing long sequence semantic association. Both GRU and LSTM outperform typical RNNs in terms of their effectiveness, and GRU has the added advantage of having a lesser computational complexity compared to LSTM. Nevertheless, GRU is unable to fully resolve the issue of gradient vanishing. Simultaneously, its function as a variation of RNN possesses a significant limitation inherent to the RNN structure. Specifically, it cannot be computed in parallel, which becomes a crucial bottleneck in the advancement of RNN as the quantity of data and model size progressively grows.

### 2.4.5 Attention Mechanism

1. *What is attention mechanism?*

An attention mechanism is a component commonly used in neural network architectures, particularly in the context of sequence-to-sequence models and neural networks dealing with sequential data [27]. It enables the model to focus on specific parts of the input sequence (or sequences) when making predictions, rather than treating all parts of the input equally. This selective attention mechanism allows the model to assign different weights or importance to different elements of the input sequence, effectively attending to the most relevant information for the task at hand. In natural language processing tasks, such as machine translation, text classification, and sentiment analysis, attention mechanisms have proven to be particularly effective.

2. *What are the components in attention mechanism?*

An attention mechanism typically includes the following components [27]:

1) ***Query, Key, and Value*:** These are the fundamental components of attention mechanisms. The *query* (*Q*) represents the current state or position in the decoding process. The *key* (*K*) and *value* (*V*) represent the information in the encoder output sequence. These components are used to compute attention scores, which determine how much attention should be paid to each element in the input sequence.

2) ***Score Calculation Function*:** This function computes the similarity or compatibility between the query and each key in the input sequence. It produces a score for each key, indicating its relevance to the current query. Common scoring functions include dot product, additive (concatenation followed by a linear transformation), and multiplicative (element-wise product followed by a linear transformation).

3) ***Attention Weights*:** These are the normalized scores obtained from the score calculation function. They represent the importance or relevance of each element in the input sequence relative to the current query. Attention weights form a probability distribution over the input sequence, indicating how much attention should be paid to each element.

4) ***Context Vector*:** The context vector is obtained by taking a weighted sum of the values in the input sequence, where the weights are given by the attention weights. It represents the relevant information from the input sequence that the model should focus on for the current query.

5) ***Attention Mechanism Types*:** There are various types of attention mechanisms, including additive attention, multiplicative attention, self-attention, and multi-head attention. Each type may have variations in its components and computation rules, but they generally share the same fundamental components described above.

These components work together to enable the model to selectively attend to relevant parts of the input sequence, allowing it to make more informed predictions or generate more accurate outputs in tasks such as machine translation, text summarization, and sentiment analysis.

*3. Common attention calculation rules*

Q and K were concatenated on the vertical axis, a linear change was made, and then the softmax processing was used to obtain the result. Finally, the tensor multiplication with V was performed.

$$Attention(Q,k,V) = Soft\max(Linear([Q,K])) \cdot V \tag{2.14}$$

Q and K are concatenated on the vertical axis, and then activated by the tanh function after a linear change, and then the internal sum is performed. Finally, the softmax processing is used to obtain the result and the tensor multiplication with V is done.

$$Attention(Q,k,V) = Soft\max(\text{sum}(\tanh(Linear([Q,K])))) \cdot V \tag{2.15}$$

It can use a dot product of Q with the transpose of K, divide by a scaling factor, use softmax to get the result, and then do a tensor multiplication with V.

$$Attention(Q,K,V) = Soft\max(\frac{Q \cdot K^T}{\sqrt{d_K}}) \cdot V \tag{2.16}$$

When the attention weight matrix and V are three-dimensional tensors, with the first dimension representing the number of batches, we perform a “batch matrix multiplication (BMM)” operation. This operation is a specific type of tensor multiplication.

4. *Types of Attention Mechanism*

There are several types of attention mechanisms commonly used in deep learning models. Each type of attention mechanism has its characteristics and is suitable for different types of tasks or architectures. Here are some of the common types of attention:

1) ***Global Attention*:** In global attention, each element in the input sequence is considered when computing attention scores. It computes the attention weights based on the similarity between the query and all elements in the input sequence. Global attention is suitable for tasks where the entire input sequence is relevant to each output element, such as machine translation.

2) ***Local Attention*:** Local attention focuses only on a subset of elements in the input sequence, rather than considering all elements. It computes the attention weights based on a window

or range of elements centered around the current output position. Local attention is useful for tasks where only a local context is relevant to each output element, such as text summarization.

3) ***Scaled Dot-Product Attention*:** Scaled dot-product attention computes attention scores as the dot product between the query and key vectors, scaled by the square root of the dimensionality of the key vectors. It is commonly used in transformer architectures and is efficient for capturing global dependencies between input and output sequences.

4) ***Additive Attention*:** Additive attention computes attention scores as a weighted sum of the concatenation of the query and key vectors passed through a learned linear transformation followed by a non-linear activation function (such as ReLU). It allows the model to learn a more complex mapping between the query and key vectors compared to dot-product attention.

5) ***Multiplicative Attention*:** Multiplicative attention computes attention scores as the cosine similarity between the query and key vectors. It is a simple attention mechanism that captures the similarity between the query and key vectors based on their directions in the vector space.

6) ***Self-Attention*:** Self-attention, also known as intra-attention or intra-modality attention, computes attention scores within the same sequence (e.g., within a sentence or document). It allows the model to capture dependencies between different elements in the input sequence and is commonly used in transformer architectures for tasks like language modeling and text generation.

7) ***Cross-Modal Attention*:** Cross-modal attention computes attention scores between different modalities (e.g., between text and image features). It enables multimodal models to selectively attend to relevant information across different modalities, facilitating tasks such as image captioning or multimodal sentiment analysis.

5. *How to implement attention mechanism?*

Implementing an attention mechanism involves several steps. Below are the general steps for implementing an attention mechanism in a deep learning model:

1) ***Define the Components:*** This stage is to identify the components required for the attention mechanism, including the query, key, and value vectors. These vectors will be used to compute attention scores.

2) ***Compute Attention Scores*****:** This stage is to calculate attention scores between the query and key vectors. This can be done using different scoring functions, such as dot product, additive, or multiplicative attention.

3) ***Compute Attention Weights*****:** This stage is to normalize the attention scores to obtain attention weights. Typically, this involves applying a softmax function to the attention scores to convert them into a probability distribution over the input sequence.

4) ***Compute the Context Vector*****:** This stage is to compute a weighted sum of the value vectors using the attention weights. This yields the context vector, which represents the attended information from the input sequence.

5) ***Incorporate the Context Vector*****:** This stage is to combine the context vector with the original input data. This can be done by concatenating or adding the context vector to the input data, depending on the specific architecture of the model.

6) ***Integrate into Model Architecture*****:** This stage is to modify the model architecture to incorporate the attention mechanism. This typically involves adding attention computation steps to the forward pass of the model, especially in layers where attention is needed (e.g., in the decoder of a sequence-to-sequence model).

7) ***Training and Optimization*****:** This stage is to train the model using backpropagation and an optimization algorithm (e.g.,

stochastic gradient descent, Adam). Monitor the model's performance on a validation set and adjust hyperparameters as needed.

8) ***Inference*:** During inference, it uses the trained model to make predictions on new data. The attention mechanism will dynamically focus on relevant parts of the input sequence, aiding in generating accurate predictions.

9) ***Evaluation and Fine-Tuning*:** It is to evaluate the performance of the model on a separate test set. Fine-tune the model and attention mechanism based on performance metrics and feedback from the evaluation.

10) ***Iterate and Improve*:** This iterates on the model architecture, attention mechanism, and training process to further improve performance. Experiment with different types of attention mechanisms and architectures can help to find the best solution for the task at hand.

### 2.4.6 Transformer Learning and BERT

2.4.6.1 Transformer Learning

Transformer learning refers to the process of training transformer architectures, a class of deep learning models that have revolutionized natural language processing (NLP) and other sequential data tasks [28]. The term "transformer" originates from the seminal paper "Attention is All You Need" by Vaswani et al., published in 2017, where the authors introduced the transformer architecture as an alternative to traditional recurrent neural networks (RNNs) and convolutional neural networks (CNNs) for sequence modeling tasks. Therefore, attention mechanisms are the core components of transformer architectures. Figure 2.8 presents a structure of transformer learning using attention mechanisms as core components.

**Self-Attention Mechanism:** Transformers utilize self-attention mechanisms to capture dependencies between different elements (or tokens) within a sequence. Self-attention allows the model to weigh the importance of each element relative to others, enabling it to focus on relevant parts of the input sequence while processing it.

**Multi-Head Attention:** Transformers employ multi-head attention, which extends the basic self-attention mechanism by performing multiple attention operations in parallel. Each attention head learns to focus on different aspects of the input sequence, providing richer representations.

**Positional Encoding:** Since self-attention mechanisms do not inherently encode the positional information of tokens in a sequence, transformers incorporate positional encodings to provide the model with information about the positions of tokens within the sequence. Positional encodings are added to the input embeddings before being passed through the attention mechanism.

**Transformer Architecture:** The transformer architecture consists of encoder and decoder layers, each of which contains multiple self-attention sub-layers followed by position-wise feedforward neural networks. These self-attention sub-layers allow the model to capture long-range dependencies and learn contextual representations efficiently.

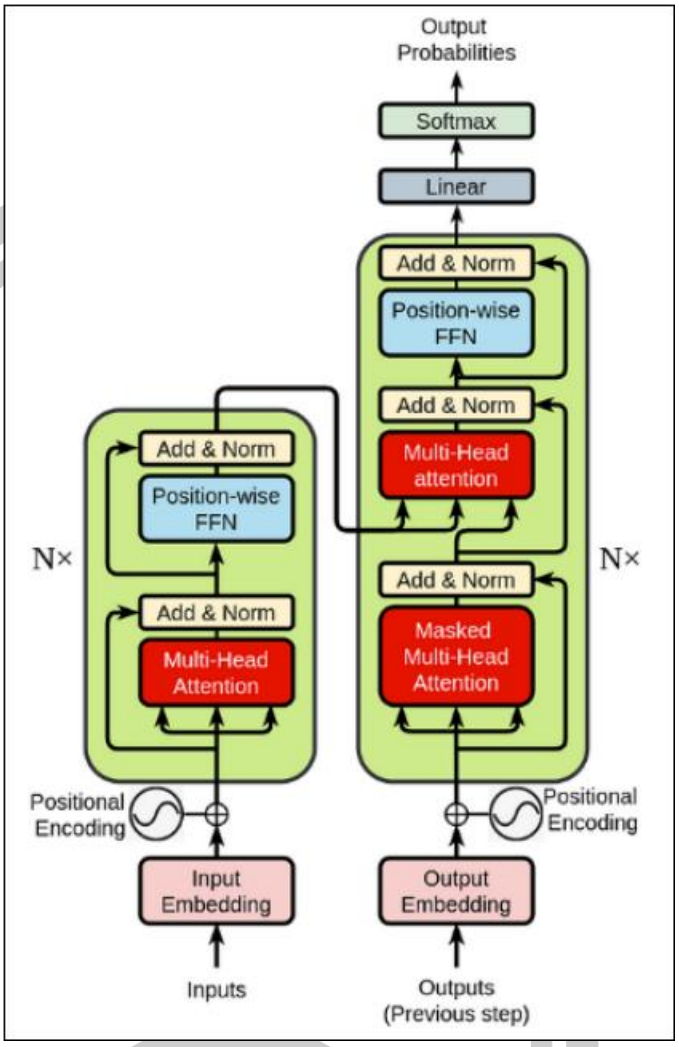



Figure 2.8 A Structure of Transformer Learning

From: https://www.siasat.com/the-transformer-model-revolutionizing-natural-language-processing-2535879/

Here is how attention is central to transformer learning:

***Efficient Parallelization*:** Attention mechanisms, especially self-attention, enable efficient parallelization during training and inference, as the computations within each attention head can be performed independently. This parallelization facilitates faster training and inference times, making transformers highly scalable.

In summary, attention mechanisms are integral to transformer learning, providing the model with the ability to capture long-range dependencies, focus on relevant information, and achieve state-of-the-art performance on a wide range of sequential data tasks.

If consider key aspects of transformer learning, they include:

***Architecture*:** Transformers consist of an encoder-decoder architecture, with each encoder and decoder layer containing multiple self-attention mechanisms followed by position-wise feedforward neural networks. The self-attention mechanism enables the model to capture long-range dependencies within input sequences efficiently.

***Self-Attention Mechanism*:** The self-attention mechanism allows transformers to weigh the importance of each element (or token) in a sequence relative to others. It enables the model to focus on relevant parts of the input sequence while processing it, capturing dependencies irrespective of the distance between tokens.

***Multi-Head Attention*:** Transformers typically employ multi-head attention, where multiple attention heads operate in parallel, each learning to focus on different aspects of the input sequence. Multi-head attention provides the model with richer representations and enables it to capture diverse patterns in the data.

***Positional Encoding*:** Since self-attention mechanisms do not inherently encode the positional information of tokens in a sequence, transformers incorporate positional encodings to provide the model with information about the positions of tokens within the sequence. Positional encodings are added to the input embeddings before being passed through the attention mechanism.

***Training*:** Transformers are trained using standard backpropagation algorithms, such as stochastic gradient descent (SGD) or Adam, with objectives tailored to the specific task at hand. Common objectives include cross-entropy loss for classification tasks and mean squared error for regression tasks.

2.4.6.2 Bidirectional Encoder Representations from Transformers (BERT)

BERT represents a significant application of transformer learning, demonstrating the effectiveness of transformer-based models in natural language processing tasks [28]. It is a specific instance of transformer architecture designed for pre-training contextualized word representations. BERT is a transformer-based deep learning model introduced by researchers at Google AI Language in the paper “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding” published in 2018. BERT has significantly advanced the state-of-the-art in natural language processing (NLP) by providing pre-trained contextualized word

representations that can be fine-tuned for a wide range of downstream tasks. The connection between transformer learning and BERT can be explained as follows:

***Transformer Architecture*:** BERT is built on top of the transformer architecture. It relies on self-attention mechanisms to capture dependencies between different elements within a sequence efficiently.

***Pre-training Strategy*:** BERT adopts a pre-training strategy where the model is first pre-trained on large corpora of text data using unsupervised objectives. During pre-training, BERT learns to predict masked words in a sentence (masked language modeling) and predict whether two sentences are consecutive or not (next sentence prediction). This pre-training process allows BERT to learn rich contextual representations of words.

***Fine-tuning*:** After pre-training, BERT can be fine-tuned on downstream tasks with task-specific labeled data. Fine-tuning involves updating the parameters of the pre-trained BERT model using supervised learning techniques (e.g., gradient descent) to adapt it to the specific task at hand, such as sentiment analysis, named entity recognition, or question answering.

***Contextualized Word Representations*:** BERT generates contextualized word representations by considering the entire input sentence bidirectionally. Unlike traditional word embeddings like Word2Vec or GloVe, which provide fixed representations for each word, BERT produces word representations that vary depending on the context in which the word appears. This enables BERT to capture nuances in meaning and syntactic structure more effectively.

2.4.6.3 How to develop BERT's pre-trained model?

Developing a pre-trained model, such as BERT [28], involves several steps, including data collection, pre-processing, model architecture design, training, evaluation, and fine-tuning. Below is a comprehensive and sequential outline for creating a pre-trained model and an architecture of BERT can be presented as Figure 2.9.

***Data Collection*:** To develop a pre-trained model, it is necessary to gather a large and diverse dataset relevant to the task you want to address, and then ensure that the dataset covers a wide range of examples and variations relevant to the task.

***Data Pre-processing*:** This stage is to pre-process the raw data to ensure it is in a format suitable for training. This may involve tasks such as tokenization, lowercasing, punctuation removal, and data cleaning to remove noise and irrelevant information.

***Model Architecture Design*:** The purpose of this stage is to choose an appropriate architecture for your specific goal. Pre-trained language models such as BERT often employ transformer layers that incorporate self-attention processes in their architecture. Additionally, it is necessary to determine the precise structure of the model, encompassing the quantity of layers, hidden units, activation functions, and other hyperparameters.

***Tokenization*:** The purpose of this stage is to convert the pre-processed data into sequences of tokens. This process involves transforming the unprocessed text data into a format that is appropriate for feeding into the chosen model. Additionally, a tokenizer that is consistent with the model's structure is utilized.

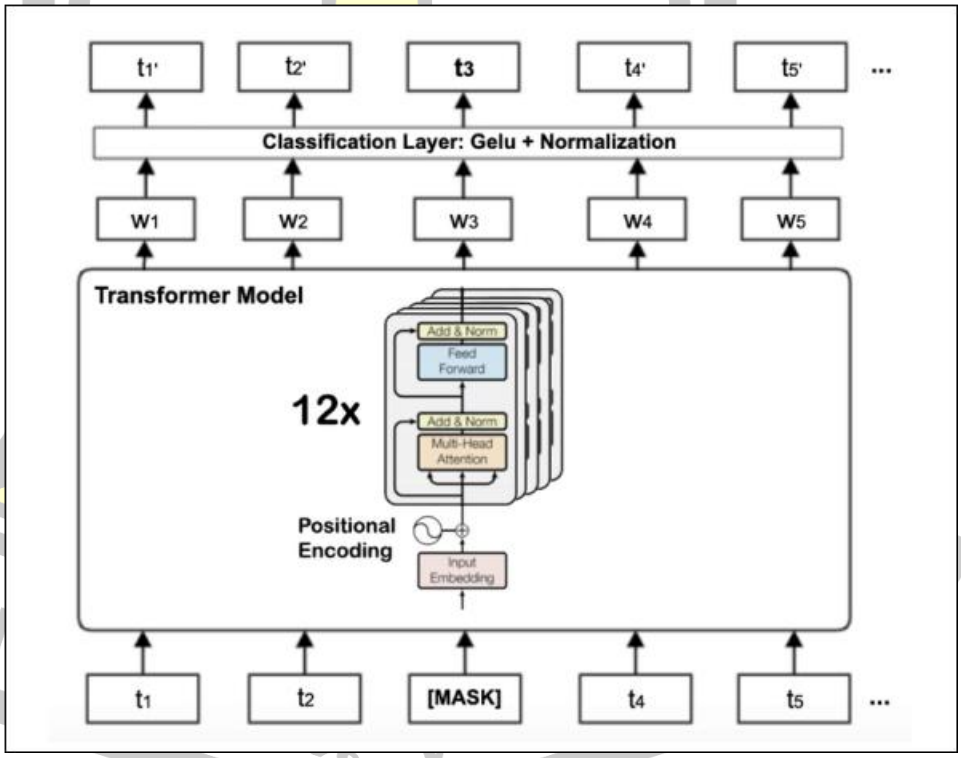



Figure 2.9 An architecture of BERT

From: https://www.researchgate.net/figure/The-Transformer-based-BERT-base-architecture-with-twelve-encoder-blocks_fig2_349546860

Model Training: The purpose of this stage is to set the starting values of the model parameters using either random weights or pre-trained weights, if they are available. Subsequently, the model undergoes training on the pre-processed data utilizing suitable optimization techniques such as stochastic gradient descent or Adam. Finally, the purpose is to oversee the training process, which involves tracking the convergence of loss, the duration of training, and the exploitation of resources.

Evaluation: In this stage, the performance and generalization ability of the trained model is evaluated by examining it on a validation dataset. Metrics related to the task at hand, including accuracy, precision, recall, F1 score, and others, are computed. Additionally, it needs to analyze the model's performance and identify areas for improvement.

2.4.6.4 How to fine-tune BERT's pre-trained model?

Fine-tuning BERT's pre-trained model adapts to a given downstream task by changing its parameters with task-specific data. Here is a step-by-step tutorial for fine-tuning BERT's pre-trained model:

Choose a Pre-Trained BERT Model: First, the process starts by choosing a pre-trained BERT model that is most suitable for your specific purpose and needs. BERT models are available in different sizes, such as base and large, and variations, such as uncased and cased. It is important to select a pre-trained model that is trained on a dataset that is relevant to your task and also fits within the computational resources you have available.

Prepare Task-Specific Data: Once a suitable pre-trained BERT model has been chosen, the next step is to gather or prepare a dataset that is specific to your particular purpose. In order to ensure that the dataset is labeled or targeted appropriately for the given task, such as classification labels or regression targets, it is necessary to preprocess the gathered dataset. This preprocessing involves tasks such as tokenization, converting to lowercase, removing punctuation, and cleaning the data.

Tokenization: Tokenization is a crucial step in the process of text pre-processing. During this stage, the text dataset is tokenized using the BERT tokenizer, which is adapted to the specific task at hand. This step converts the raw text data into a format suitable for input into the pre-trained BERT model. The BERT tokenizer converts text into a sequence of token IDs, and then add special tokens to the tokenized sequence, such as [CLS] (classification), [SEP] (separator), and [MASK]

(mask). These tokens help BERT understand the structure of the input sequence. Subsequently, it is required to randomly mask a specific proportion of tokens in the input sequence. Usually, a predetermined proportion (e.g., 15%) of tokens is selected for the purpose of masking. Ultimately, it substitutes the chosen tokens with the [MASK] token. This stage emulates the situation in which the model is required to predict masked tokens by analyzing the context surrounding them.

Define Model Architecture: The purpose of this stage is to determine the structure of the neural network, which consists of the pre-trained BERT model as a foundation, along with supplementary layers for task-specific processing such as classification or regression. In addition, this allows for the selective freezing of certain layers in the pre-trained BERT model to prevent them from being modified during the fine-tuning process. This is particularly useful when there is a scarcity of training data or restricted processing resources.

Fine-Tuning Procedure: This stage involves initializing the model parameters using the pre-trained BERT weights. In addition, it requires setting up the optimizer (such as Adam) and selecting an appropriate loss function for your specific assignment. Afterwards, the pre-trained BERT model is optimized for the task-specific dataset by updating its parameters through the process of backpropagation and gradient descent. Next, it is necessary to proceed through the dataset for several epochs. Also, it is necessary to oversee the training process, which includes monitoring the convergence of loss, training duration, and resource consumption. Fine-tuning a pre-trained BERT model involves setting various hyperparameters that govern the training process. These hyperparameters include learning rate, batch size, number of epochs, dropout rate, optimizer, and more. The hyperparameters that need to be configured are presented in Table 2.1.

Validation: Once the pre-trained BERT has been adjusted, it is necessary to evaluate the performance of the fine-tuned model by evaluating it on a different validation dataset. This evaluation helps to minimize overfitting. One can accomplish this by computing assessment metrics like as accuracy, precision, recall, F1 score, or other metrics that are relevant to the specific task at hand. If the fine-tuned model yields unsatisfactory outcomes, it is necessary to readjust hyperparameters, such as the learning rate, batch size, and dropout rate, according on the validation results.

Table 2.1 The hyperparameters configuration of BERT

| Hyperparameters | Details of Fine-tuning |
|---|---|
| Learning Rate | A common starting point is to use a learning rate of 5e-5 or 3e-5. |
| Batch Size | Common batch sizes for fine-tuning BERT range from 16 to 64 and adjust based on empirical observations and performance on a validation dataset during training. |
| Number of Epochs | The number of epochs for fine-tuning BERT for sentiment classification can vary depending on factors such as the size of the dataset. As a starting point, you can try training BERT for sentiment classification with a few epochs (e.g., 3-5 epochs) and gradually increase the number of epochs if needed while monitoring the validation loss. |
| Optimizer | Adam |
| Dropout Rate | A common starting point for the dropout rate is around 0.1 to 0.5. This range has been shown to work well for many tasks and architectures, including fine-tuning BERT. |

***Testing*:** After the process of fine-tuning is finished, it is necessary to examine the performance of the fine-tuned model on a distinct test dataset in order to evaluate its effectiveness on data that it has not previously seen. It is capable of computing evaluation metrics to quantify the model's efficacy in solving the given task.

### 2.4.7 Text Representation

Text representation [29] involves the conversion of textual input into a numerical format that may be comprehended and processed by machine learning algorithms or other computer models. Text representation plays a crucial role in natural language processing (NLP) tasks, where algorithms rely on text data to carry out tasks like classification, clustering, sentiment analysis, translation, and others. There exist several methods for expressing textual information, each possessing its own set of benefits and drawbacks. Several prevalent ways for representing text include:

*One-Hot Encoding* [30]: One-hot encoding represents each word in a vocabulary as a binary vector, where each dimension corresponds to a unique word in the vocabulary. The vector is all zeros except for the index corresponding to the word,

which is set to 1. Suppose we have a vocabulary of [“*cat*”, “*dog*”, “*bird*”]. The one-hot encoding for “*dog*” would be [0, 1, 0] because “*dog*” is the second word in the vocabulary. This technique is simple and easy to implement but results in high-dimensional, sparse vectors, making it inefficient for large vocabularies.

*Bag-of-Words* (*BoW*) By 2023, Zhu et al [31]: Bag-of-Words represents each document as a vector, where each dimension corresponds to a unique word in the vocabulary, and the value of each dimension represents the frequency of that word in the document. Suppose we have two documents: “*I love cats*” and “*I hate dogs*”. The BoW representation for these documents are [1, 1, 0, 0] and [1, 0, 1, 1] respectively, where the dimensions correspond to [“*I*”, “*love*”, “*cats*”, “*hate*”,“*dogs*”]. In general, BoW captures the frequency of words in documents but disregards the order and context of words, which may lead to loss of information.

*Word Embeddings* [32]: Word embeddings represent words as dense, low-dimensional vectors in a continuous vector space, where words with similar meanings are closer together. Word embeddings are learned from large text corpora using techniques like Word2Vec, GloVe, or BERT. Word embeddings capture semantic relationships between words, such that similar words have similar vector representations. For example, the vectors for “cat” and “dog” might be close together in the embedding space because they are both related to pets. Word embeddings capture semantic meaning and contextual relationships between words, making them suitable for various NLP tasks such as sentiment analysis, named entity recognition, and machine translation. Although numerous word embedding techniques exist, Word2Vec and GloVe are two of the most frequently employed:

*Word2Vec* [33]: Word2Vec is a shallow neural network model trained to predict words within a context window given a target word or vice versa. It learns distributed representations of words based on the contexts in which they appear in the training corpus. This technique typically consists of two main architectures: Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW predicts the target word based on the surrounding context words, while Skip-gram predicts context words given a target word. It is trained on large text corpora using stochastic gradient descent or negative sampling to optimize the prediction task. Word2Vec embeddings capture syntactic and semantic relationships between words, making them suitable for tasks such as word similarity, analogy completion, and sentiment analysis.

Global Vectors for Word Representation (GloVe) [34]: GloVe is a global matrix factorization technique that leverages co-occurrence statistics from a corpus to learn word embeddings. It constructs a co-occurrence matrix of word-word co-occurrences and factorizes it into low-dimensional word vectors. GloVe training involves minimizing a loss function that measures the difference between the dot product of word vectors and the logarithm of the word co-occurrence probabilities. GloVe embeddings capture global word co-occurrence statistics, which helps in preserving both semantic and syntactic relationships between words. They are effective for tasks such as word analogy completion, named entity recognition, and sentiment analysis.

Both Word2Vec and GloVe embeddings have been extensively utilized and own their own merits and drawbacks. The selection between them frequently relies on the particular demands of the NLP assignment, the magnitude of the training corpus, and the computational resources accessible for training and inference. Transformer-based models such as BERT have become increasingly popular for learning word embeddings. These models capture contextual information in both directions, resulting in excellent performance in a range of NLP applications.

## 2.5 Evaluation Metrics

1.Confusion Matrix

The confusion matrix [35], As shown in Figure 2.10, sometimes referred to as the likelihood matrix or the error matrix, is a fundamental component in the field of machine learning. Confusion matrices serve as visual aids, primarily in the context of supervised learning, whereas in unsupervised learning they are commonly referred to as matching matrices. Evaluation mostly involves comparing the classification outcomes with the actual measured values, and the accuracy of the classification findings can be visually represented using a confusion matrix. The structure of the confusion matrix is typically shown as follows.

Each column in the confusion matrix corresponds to the expected class, and the sum of each column reflects the count of data predicted to belong to that class. Each row corresponds to the actual classification class of the data, and the total count of data in each row shows the number of occurrences in that class; the value in each column indicates the number of correctly predicted data belonging to that class.

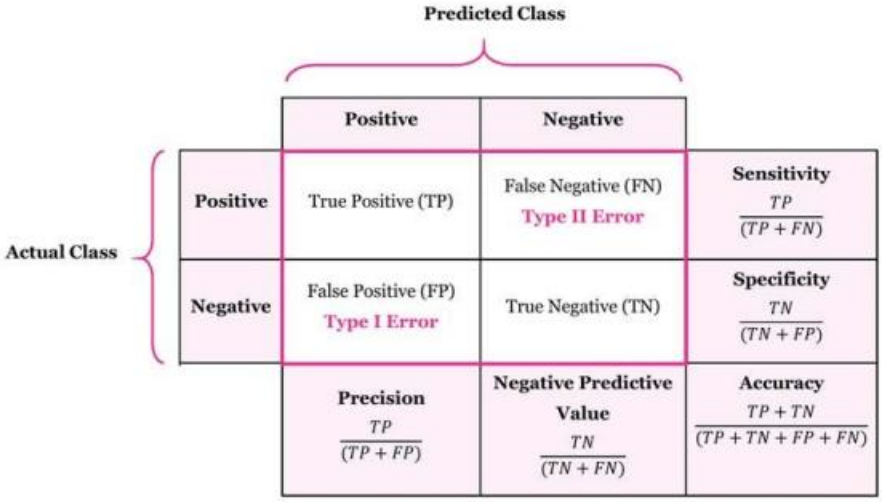



Figure 2.10 Confusion Matrix

From: https://encord.com/glossary/confusion-matrix/

*True Positive* (*TP*) [36]: It refers to an outcome where the model correctly predicts the positive class. Simply speaking, it indicates that the model accurately identified an instance as belonging to a certain category when it indeed should have.

*False Negative* (*FN*) [37]: It refers to an outcome where the model incorrectly predicts the negative class for an instance that actually belongs to the positive class. It represents a mistake where the model fails to identify an instance as positive when it truly is, effectively missing a true positive case.

*False Positive* (*FP*) [38]: It refers to an outcome where the model incorrectly predicts the positive class for an instance that actually belongs to the negative class. This is essentially a type of error where the model has identified something as true or positive (according to the classification task) when it is not.

*True Negative* (*TN*) [39]: It refers to an outcome where the model correctly predicts the negative class for an instance that actually belongs to the negative class. This means that the model accurately identifies an instance as not belonging to the category (or categories) of interest.

2. Accuracy (Acc)

The metric of accuracy is often employed as a classification performance measure. The accuracy of a model may be measured by calculating the ratio of correct identifications made by the model to the total number of samples. Typically, a model's quality improves as its accuracy increases. The formula can be written as (2.17).

$$Acc = \frac{TP + TN}{TP + FN + TN + FP} \quad (2.17)$$

3. Precision (P)

The precision ratio, sometimes referred to as the positive predictive value, is the proportion of samples classified as positive by the model that are truly positive. Typically, a higher precision rate indicates a superior model. The formula can be written as (2.18).

$$P = \frac{TP}{TP + FP} \quad (2.18)$$

*4. Recall (R)*

Recall is a measure that quantifies the proportion of positive class samples correctly recognized by the model, relative to the total number of positive class samples. Typically, higher recall indicates that the model accurately predicts more positive class samples, indicating a better performance of the model. The formula can be written as (2.19).

$$R = \frac{TP}{TP + FN} \quad (2.19)$$

5. The Receiver Operating Characteristic (ROC)

The ROC curve [40] is a graphical representation used to evaluate the performance of a binary classifier system as its discrimination threshold is varied. It plots two parameters:

*True Positive Rate* (*TPR*): Also known as sensitivity or recall, it measures the proportion of actual positives that are correctly identified by the model. It is calculated as TPR = TP / (TP + FN), where TP is the number of true positives and FN is the number of false negatives.

*False Positive Rate* (*FPR*): It measures the proportion of actual negatives that are incorrectly identified as positives by the model. It is calculated as FPR = FP /

(FP + TN), where FP is the number of false positives and TN is the number of true negatives.

The ROC curve plots TPR against FPR at various threshold settings. The threshold refers to the probability (or some other measure) at which the classification decision changes from one class to another. Adjusting the threshold affects the classifier's sensitivity and specificity, and thus changes the FPR and TPR values.

In Figure 2.11, the ROC curve is plotted with TPR against the FPR where TPR is on y-axis and FPR is on the x-axis.

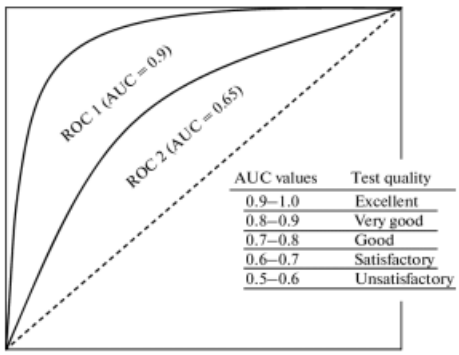



Figure 2.11 ROC and AUC
https://www.researchgate.net/figure/An-example-of-ROC-curves-with-good-AUC-09-and-satisfactory-AUC-065-parameters_fig2_276079439

6. The Area Under the Curve (AUC)

The AUC [41] specifically refers to the area under the ROC curve, a popular evaluation metric used in binary classification to understand a model's diagnostic ability. The AUC represents a probability measure of a classifier's ability to distinguish between the classes and is used to quantify the overall performance of a classification model. Key aspects of AUC are:

*Value Range*: The AUC ranges from 0 to 1, where an AUC of 1 indicates a perfect model that can completely distinguish between positive and negative classes. An AUC of 0.5 suggests a model with no discriminative ability, equivalent to random guessing. In practice, an AUC below 0.5 indicates a model performing worse than

random guessing, but this situation typically leads to inverting the model's predictions to improve performance.

*Interpretation*: A higher AUC value means that the model has a better performance in distinguishing between the positive and negative classes across all possible thresholds. It integrates the model's performance across all classification thresholds, making it less sensitive to changes in the decision threshold than other metrics like accuracy.

## 2.6 Related Work

In recent years, multimodal sentiment analysis has garnered significant attention from researchers [42-44]. The concept of "emotion" is broad yet somewhat ambiguous. In the literature, terms such as feeling, emotion, sentiment, and opinion are often used interchangeably to describe various aspects of emotion. To address this ambiguity, Hovy et al. [45] from Carnegie Mellon University proposed a consistent definition and explanation of these concepts. They argue that "emotion" not only refers to a specific subjective feeling but also encompasses a wide range of human experiences, including sensory, physical, psychological, and spiritual states. These emotions can be conveyed through language, facial expressions, physiological signals, and other means of communication.

Sentiment analysis plays a crucial and indispensable role in artificial intelligence, particularly as natural language processing (NLP) continues to advance. It has become an essential technology in the pursuit of AI-driven objectives, enabling intelligent systems to perceive, interpret, and understand human emotions [46]. Despite the complexity and diversity of research challenges in sentiment analysis, studies in this field generally fall into two main categories: fine-grained emotion classification and coarse-grained sentiment polarity detection. Sentiment polarity reflects a subject's attitude toward a given object, typically classified as positive, neutral, or negative. In contrast, emotion classification identifies specific affective states, such as love, sorrow, or anger. Unlike emotions, which can be fluid and context-dependent, sentiment polarity is often more stable, as it is influenced by particular objects or entities with consistent emotional associations.

Early sentiment analysis research primarily focused on either text or images, relying on traditional machine learning classification algorithms such as K-nearest neighbors (KNN), support vector machines (SVM), maximum entropy classifiers, and

Bayesian classifiers. However, with the rise of deep learning, researchers have increasingly adopted deep neural networks to extract feature representations from text and images for sentiment analysis, achieving remarkable performance. Despite these advancements, relying on a single modality presents limitations. Text-based analysis can be affected by ambiguous semantics, while image-based analysis may struggle with unclear visual representations. To overcome these challenges, researchers have explored image-text information fusion, leveraging the complementary nature of multimodal data. With the rapid advancement of multimedia technology, this fusion approach has emerged as a key area of contemporary research [47] .

### 2.6.1 Multimodal Sentiment Analysis

In modern society, social networks are predominantly multimodal, with text and images being the most common forms of communication. These two modalities are highly interrelated and complementary, reinforcing and enhancing one another to create engaging, dynamic, and immersive social experiences. Research on multimodal sentiment analysis dates back to 2010. Zontone et al. [48] proposed leveraging a photo's textual content for automatic annotation and then using the sentiment lexicon SentiWordNet to predict the sentiment of the corresponding image. In China, related research began in 2011 when Yang Feng et al. [49] from Northeastern University introduced an early model that combined sentiment analysis with automated data crawling for microblog streams. Their approach involved extracting and analyzing image-text pairs from Sina Weibo, as well as constructing sentiment lexicons. A key challenge in image-text sentiment analysis lies in feature extraction—the process of identifying meaningful representations from both text and images. The integration of multimodal data, known as cross-modal fusion, plays a crucial role in combining information effectively, enabling the construction of joint feature representations and facilitating sentiment analysis.

During the data preprocessing stage, image data typically undergoes steps such as resizing, normalization, and feature extraction, while text data requires tokenization, stop word removal, and word vectorization. In terms of algorithm selection, researchers often employ convolutional neural networks (CNNs) to extract image features and use word embedding models (such as Word2Vec or BERT) to process text data. Additionally, cross-modal fusion techniques (such as attention mechanisms or multimodal neural networks) are widely applied to integrate image and text information, enabling more accurate sentiment analysis.

*1. Related Work on Feature Extraction Techniques in Multimodal Sentiment Analysis*

Chen et al. [50] conducted a correlation analysis on image-text data from Twitter and found that co-occurring image-text modalities in social media not only exhibit semantic relationships at the entity level but may also demonstrate emotional consistency at lower or intermediate levels of visual representation. For instance, when users share personal experiences on Facebook, they often convey happiness or sadness through images with warm or cool color tones.

Joshi et al. [51] shared a similar perspective, emphasizing the fundamental difference between image recognition and image sentiment analysis. While image recognition primarily relies on low-level visual features, sentiment analysis requires extracting high-level elements to accurately interpret both the semantic meaning and emotional tone of an image.for instance, an image containing "blue sky and white clouds" might be classified as "sky" or "landscape" in image recognition, while in sentiment analysis, it could be interpreted as "joyful" or "peaceful." During the data preprocessing stage, image data typically undergoes steps such as resizing, normalization, and feature extraction, while text data requires tokenization, stop word removal, and word vectorization. In terms of algorithm selection, researchers often employ convolutional neural networks (CNNs) to extract image features and use word embedding models (such as Word2Vec or BERT) to process text data. These preprocessing steps and algorithm choices provide a crucial foundation for subsequent methodological design decisions, such as the application of cross-modal fusion techniques to better integrate image and text information for more accurate sentiment analysis.

In 2013, Borth et al. [52] introduced the image-character combination method, which uses adjective-noun pairs (ANPs) to describe image content. Examples of ANPs include "beautiful scenery," "delicious food," and "dark night." They also developed the Visual Sentiment Ontology (VSO), a corpus containing over 1,200 ANPs, designed to bridge the gap between low-level visual features and high-level emotional semantics. The VSO has since inspired further research and has gained significant attention, with scholars expanding upon its foundation.

While the Adjective-Noun Pair (ANP) approach provides semantic and emotional insights for sentiment analysis, it relies on manually defined rules and annotations, resulting in limited generalization ability. In contrast, deep learning offers

robust feature representation, making transfer learning with pre-trained neural networks the standard technique in modern visual sentiment analysis. In 2015, You et al. [53] introduced the Progressive Convolutional Neural Network (PCNN) for visual sentiment analysis. Their model was pre-trained on Flickr's large dataset and later fine-tuned using transfer learning on a new Twitter dataset. Similarly, Ortis et al. [54] reviewed recent research on image sentiment analysis, analyzing its evolution over the past decade. By 2023, Zhu et al. [55] highlighted multimodal sentiment analysis as a promising approach that leverages complementary information sources and consistently outperforms single-modal methods. As methodologies have advanced, researchers have moved beyond simply extracting image feature representations to integrating multiple modalities, creating a more comprehensive foundation for sentiment analysis. Ultimately, whether focusing on text-based or image-based sentiment analysis, both fields are converging toward a unified development framework that incorporates multimodal insights for improved accuracy and depth.

*2. Related Work on Modal Fusion in Multimodal Sentiment Analysis*

Multi-modal image-text fusion aims to integrate information from both images and text into a unified system for further processing and analysis. In the early stages of image-text sentiment analysis research, direct concatenation was the most commonly used fusion method.

In 2015, You et al. [56] introduced a method that combined image and text features through concatenation to perform cross-modal sentiment classification. They extracted image features using a Convolutional Neural Network (CNN) pre-trained on the ImageNet dataset, while text features were obtained using the Word2Vec word embedding model.

Building on this approach, Chen et al. [57] enhanced text feature extraction in 2017 by utilizing the TextCNN (Text Convolutional Neural Network) model to capture sequential text features. These features were then fused with image representations through splicing. Similarly, Hu et al. [58] adopted a Long Short-Term Memory (LSTM) network for extracting textual features from GloVe word vectors, while image features were derived from a pre-trained CNN model.

Xu et al. [59] observed that simple merging operations failed to capture the deeper interactive connections between text and image modalities. To address this limitation, they extracted both local and global visual features from images and applied an attention mechanism to compute bidirectional attention between text and

images. Additionally, they synchronized the semantic information of text entity nouns with corresponding local regions in the image, achieving hierarchical cross-modal integration.

Yu et al. [60] argued that, in certain application scenarios, the semantic information conveyed by images might be insufficient, whereas textual content can more explicitly express affective attitudes. To overcome this, they used visual information as a supplementary feature, applied one-way attention from text to image, and incorporated image features into text representations using a multi-head attention mechanism. They then extracted contextual information from the text for sentiment classification using a self-attention mechanism.

Expanding on this approach, Truong et al. [61] explored scenarios involving a single text accompanied by multiple images. They introduced a visual aspect attention network that leveraged image information alongside text semantics to identify emotionally charged words within the text. The goal was to extract more precise emotional cues for sentiment classification.

As a result, modern cross-modal fusion techniques have evolved beyond basic information integration and concatenation. Instead, they selectively construct unified feature representations by analyzing the impact of image-text modalities on the overall system. These techniques adapt to practical challenges and application environments by identifying meaningful correlations between modalities and selecting the most relevant data for enhanced sentiment classification.

### 2.6.2 Image-Text Sentiment Analysis Datasets

Multimodal data can be applied to a wide range of datasets, including:

1.Flickr dataset - The Flickr dataset is used for image identification and is annotated with positive, negative, and neutral parts of speech [62]. Flickr provides an Application Programming Interface (API) that allows retrieval of relevant metadata, such as image descriptions, upload dates, and tags, using the image ID. The publicly available dataset includes over 60,000 images along with their accompanying metadata. Additionally, the Flickr30k dataset contains 31,783 images, each annotated with five descriptive sentences, resulting in a total of 158,915 sentences.

2.VCGI dataset - VCGI data centers construct datasets from Chinese visual platforms using distinct affective keywords [63]. The VCGI dataset consists of 38,363 images, with specific subsets labeled using emotional keywords. To curate the dataset,

a total of 3,244 and 244 images were categorized based on emotional keywords. Additionally, from the textual data in the VCGI dataset, 300 adjective-noun pairs (ANPs) were randomly selected as emotional keywords. This process resulted in the collection of 37,158 images, ensuring the accuracy of the dataset.

3.MVSO dataset - Similar to the VSO dataset, the MVSO dataset [52] primarily collects data from Yahoo, a widely used social multimedia platform, to generate the Multilingual Visual Sentiment Ontology (MVSO). This dataset covers twelve primary languages: Arabic, Chinese, Dutch, English, French, German, Italian, Persian, Polish, Russian, Spanish, and Turkish. MVSO consists of 15,600 concepts, most of which are closely associated with the emotions conveyed in images. These concepts are predominantly defined using adjective-noun pairs (ANPs), with keywords selected from ANPs that have a sentiment score above 1. From social networking sites, 75,516 images, along with their corresponding titles, descriptions, and keywords, were collected. The English subset of the dataset is referred to as MVSOEN.

4.MVSA dataset - The MVSA dataset is a benchmark dataset for Multi-View Sentiment Analysis. The MVSA (Multivariate Sentiment Analysis) dataset [64] was created using Twitter's public streaming API (Twitter4J) to collect a diverse set of tweets. A total of 406 emotion-related terms were used to refine and filter the tweets. The dataset primarily consists of manually annotated image-text pairs sourced from Twitter, making it a valuable resource for comparing single-view and multi-view sentiment analysis. The MVSA dataset is divided into two distinct subsets:

(1)MVSA-Single – This subset contains 5,129 image-text pairs, each annotated with a sentiment label classified into three categories: positive, negative, or neutral.

(2)MVSA-Multi – This subset includes 19,600 image-text pairs. Each data point has been annotated by three independent annotators, and a voting approach was applied to consolidate these annotations into a final sentiment label.

The MVSA dataset serves as a robust benchmark for evaluating sentiment analysis models across different perspectives.

5. Yelp dataset –In [65], this study utilizes a dataset of online reviews on food and restaurants sourced from Yelp.com. The dataset focuses on five major U.S. cities: Boston (BO), Chicago (CH), Los Angeles (LA), New York (NY), and San

Francisco (SF). Among these cities, Los Angeles has the highest number of reviews, as well as the most extensive textual and visual content, whereas Boston has the fewest reviews. However, in terms of sentence count (#s) and word count (#w), the document lengths across the five cities are relatively similar. The dataset comprises nearly 44,000 reviews and approximately 244,000 images, with each data entry containing a minimum of three images.

6. Multi-ZOL dataset – The Multi-ZOL dataset compiles text-based information and reviews on mobile phones from the commercial website ZOL.com [66]. The initial dataset contains 12,587 reviews, with 7,359 being single-modal and 5,288 being multi-modal. These reviews cover 114 different brands and 1,318 mobile phone models. Each instance in the Multi-ZOL dataset consists of textual content, a set of images, and at least one—up to a maximum of six—evaluation aspects. The six aspects include cost performance, performance configuration, battery life, appearance and feel, shooting effect, and screen quality. In total, 28,469 aspect evaluations were collected, with each aspect assigned an emotion value ranging from 1 to 10. Additionally, the Twitter-15 and Twitter-17 datasets serve as examples of multi-modal datasets that incorporate both textual content and associated images. These datasets are annotated with the target entity and the sentiment polarity expressed in both text and visuals. The Twitter-15 dataset consists of 5,338 tweets with images, while the Twitter-17 dataset includes 5,972 such tweets. The sentiment of tweets in these datasets is classified into three categories [67].

7. Text and emoji-based Twitter dataset - The primary data source is the Amazon review dataset developed by Prettenhofer and Stein [68]. This dataset includes reviews in four languages—English, Japanese, French, and German—serving as representative samples for each language. It consists of 1,000 positive reviews and 1,000 negative reviews per language and domain. Additionally, tweets containing emojis are collected and used to generate phrase representations based on emoji usage. To facilitate emoji prediction, tweets are gathered for each language, considering only the top 64 most frequently used emojis. Each unique emoji is assigned a distinct label, making this a single-label classification task for emoji prediction.

8. Task-4 dataset - The Amazon review dataset by Prettenhofer and Stein [69, 70] serves as the primary data source. It includes reviews in four languages—English, Japanese, French, and German—with each language and domain containing 1,000 positive and 1,000 negative reviews. Additionally, tweets containing emojis are

collected and used to develop emoji-based phrase representations. For each language, tweets featuring the top 64 most frequently used emojis are extracted, with each unique emoji assigned an independent label. This approach simplifies emoji prediction into a single-label classification problem.

9.Product image-text Sentiment datasets –Several image-text sentiment datasets related to products can be described as follows.

1)ProductT2.0 dataset: The Institute of Computing Technology at the Chinese Academy of Sciences provided a multi-modal product review dataset containing images, text, and sentiment annotations.

2)Product Modality dataset: The University of Melbourne, Australia, provided a multi-modal product review dataset with annotations for images, text, and sentiment.

3)Product-18K dataset: The University of Tokyo, Japan, provided a multi-modal product review dataset with annotations for images, text, and sentiment.

4)HAHGA-Web dataset: Tsinghua University provided a multi-modal product review dataset with annotations for images, text, and sentiment.

5)Expedia Product Reviews dataset: Expedia offers a multi-modal product review dataset that includes photos, text, and sentiment scores.

6)TrustYou dataset: TrustYou Travel Technology developed a multi-modal product review dataset with annotations for images, text, and sentiment tags.

7)H-Mart dataset: Nagoya University, Japan, provided a multi-modal product review dataset that includes images, text, and sentiment evaluations.

8)T2-HO dataset: Nanyang Technological University, Singapore, provides a product review dataset with annotations for images, text, and sentiment polarity.

9)Yelp Product Photo dataset: Yelp provides product images along with emotion-based annotations.

10)Product Image Dataset with Multimodal Aspects (HIMAvA): The Fraunhofer Institute for Applied Research in Germany developed a dataset containing product images, text, and emotion-based comments.

11)Product Scene dataset: Wanchai Polytechnic University in Hong Kong, China, provides a dataset of product scene images along with emotion-based annotations.

12)MIMOSA Product Reviews Dataset: The Technical University of Barcelona created a dataset containing images, text, and audio from product reviews for sentiment analysis and multimodal sentiment interpretation.

### 2.6.3 Related Work on Multimodal Sentiment Analysis

Ringki Das et al. [71] conducted a study titled 'Multimodal Sentiment Analysis: A Survey of Methods, Trends, and Challenges,' which references 190 works. The study's framework, illustrated in Figure 2.12, provides a comprehensive overview of methodologies, applications, challenges, and resources related to various sentiment analysis techniques. The classification system explores the technological aspects, datasets, and evaluation metrics across different sentiment analysis approaches, including text, visual, audio, and multimodal sentiment analysis. The study highlights the transition from single-modal to multimodal sentiment analysis and emphasizes the significant potential for advancements in this field.

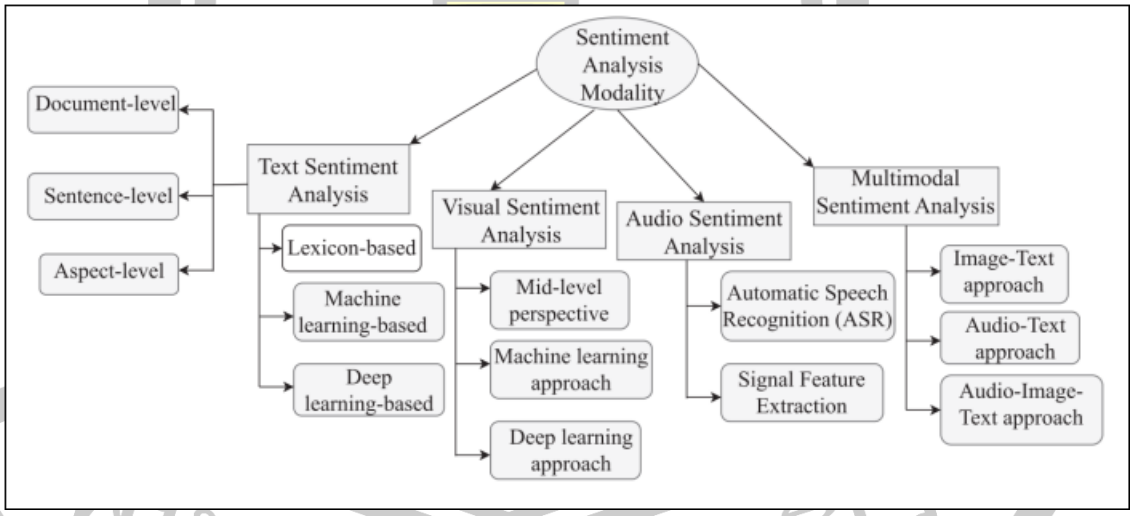



Figure 2.12 Modalities of sentiment analysis
From: https://dl.acm.org/doi/10.1145/3586075

Ankita Gandhi et al. [72] examined the advantages and disadvantages of 30 multimodal sentiment analysis (MSA) fusion techniques. These include early fusion (feature-level fusion), late fusion (decision-level fusion), hybrid fusion, model-level fusion, tensor fusion, hierarchical fusion, dual-modal fusion, attention mechanism-

based fusion, quantum-based fusion, word-level fusion, and the latest variable multimodal sentiment analysis model, as illustrated in Figure 2.13. The study highlights ten key datasets: MOSI, CMU-MOSEI, MELD, Memory Analysis Dataset, CH-SIMS, CMU-MOSEAS, MuSe-CaR, B-T4SA, FACTIFY, and MEMOTION 2. Additionally, it provides an in-depth discussion of the application domains, challenges, and potential scope of MSA.

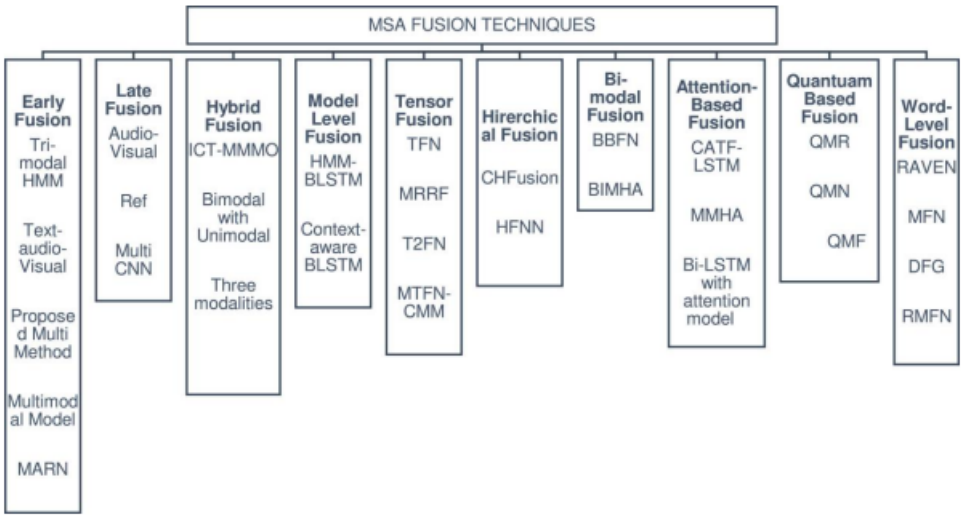



Figure 2.13 Multimodal fusion models for multimodal sentiment analysis
From:https://www.sciencedirect.com/science/article/abs/pii/S1566253522001634

Songning Lai [73] conducted a comprehensive evaluation of multimodal sentiment analysis, assessing the strengths and weaknesses of various datasets, including IEMOCAP, DEAP, CMU-MOSI, CMU-MOSEI, MELD, Multi-ZOL, CH-SIMS, CMU-MOSEAS, FACTIFY, and MEMOTION. The study explores different multimodal fusion approaches, such as early feature-based methods, medium-term model-based methods, and late-stage decision fusion methods based on emotion integration. Additionally, it provides an in-depth analysis of MultiSentiNet-Att, DFF-ATMF, AHRM, and 13 other state-of-the-art sentiment analysis models. The DFF-ATMF and MAG-BERT models are also examined in detail.

### 2.6.4 Technical Work on Multimodal Sentiment Analysis

Traditional approaches to emotion analysis primarily rely on feature-based machine learning algorithms and sentiment dictionary methods, such as syntactic

parsing labels, Part-of-Speech (POS) tags, and sentiment lexicons. However, these conventional methods depend heavily on manual feature engineering, leading to a substantial workload. To overcome these limitations, researchers have increasingly adopted deep learning models. Huang et al. [74] introduced a dynamic attention model that integrates audio and text modalities by leveraging semantic associations between them, resulting in more accurate predictions. Wen et al. [75] proposed a cross-modal context-gated convolutional network that effectively captures local cross-modal interactions, mitigates dislocation issues, and reduces noise interference. This approach offers new possibilities for designing layers in multimodal sequence modeling. Tang et al. [76] developed LSTM-based models that specifically focus on two key elements, independently modeling the top-left and upper-right contextual environments. In addition, there is a survey of multimodal sentiment analysis the can be presented Table 2.2.

Zuhe Li et al. [77] proposed an interaction-based technique for multimodal sentiment analysis using interactive transformers and soft mappings. The model consists of two key layers: the Interactive Transformer (IT) and the Soft Mapping (SM) layer. The Interactive Transformer layer is composed of N stacked blocks, each containing an Interactive Multi-Head Gated Attention (IMHGA) structure and a Feedforward Neural Network (FNN). Figure 2.12 illustrates the framework of the Soft Mapping layer, which consists of stacked Soft Attention (SA) modules and their outputs. The model is validated using the CMU-MOSEI dataset and evaluated on the MELD dataset, demonstrating competitive results. However, a limitation of this approach is that it only incorporates verbal and acoustic modalities for analysis. Traditional feature fusion methods can generally be categorized into two primary groups: backpropagation-based techniques and geometric feature space approaches. These methods often struggle to effectively regulate the transfer of information from the initial input to the fused representation. As a result, they risk losing essential information while also introducing unintended noise from each modality.

Table 2.2 Common sentiment analysis datasets

From: https://arxiv.org/abs/2305.07611

| Name | Year | Modalities | Source | Language | Number |
|---|---|---|---|---|---|
| IEMOCAP | 2008 | A+V+T | N/A | English | 10039 |
| DEAP | 2011 | A+V+T<br>A+V+T<br>A+V+T | N/A | English | 10039 |
| CMU-MOSI | 2016 | A+V+T | N/A | English | 2199 |
| CMU-MOSEI | 2018 | A+V+T | N/A | English | 23453 |
| MELD | 2019 | A+V+T | N/A | English | 13000 |
| Multi-ZOL | 2019 | V+T | ZOL.com | Chinese | 5288 |
| CH-SIMS | 2020 | A+V+T | N/A | Chinese | 2281 |
| CMU-MOSEAS | 2021 | A+V+T | YouTube | Spanish<br>Portuguese<br>German<br>French | 40000 |
| FACTLIY | 2022 | V+T | Twitter | English | 50000 |
| MEMOTION | 2022 | V+T | Reddit<br>Facebok | English | 10000 |

Wei Han et al. [78] improved the integration of multiple modalities for sentiment analysis by employing hierarchical mutual information maximization. This approach ensures that critical task-related information is preserved by maximizing mutual information at both the input and fusion stages. The overall framework of the model is illustrated in Figure 2.14. The input consists of three modalities—text, video, and speech—while the output, denoted as y, represents emotional intensity. Initially, raw data is converted into numerical sequence vectors using feature extractors for video and speech, while text is processed using markers. The model is then divided into two key components: feature fusion and maximal mutual information. To address the inefficiencies, slow processing speed, long training duration, and limited impact of image feature extraction in Vision and Language Pre-training (VLP), Wonjae Kim et al. introduced a simplified VLP model called the Visual and Language Converter (ViLT). This model significantly reduces the complexity of processing visual

information, aligning it with the efficiency of text data processing using convolution-free architectures.

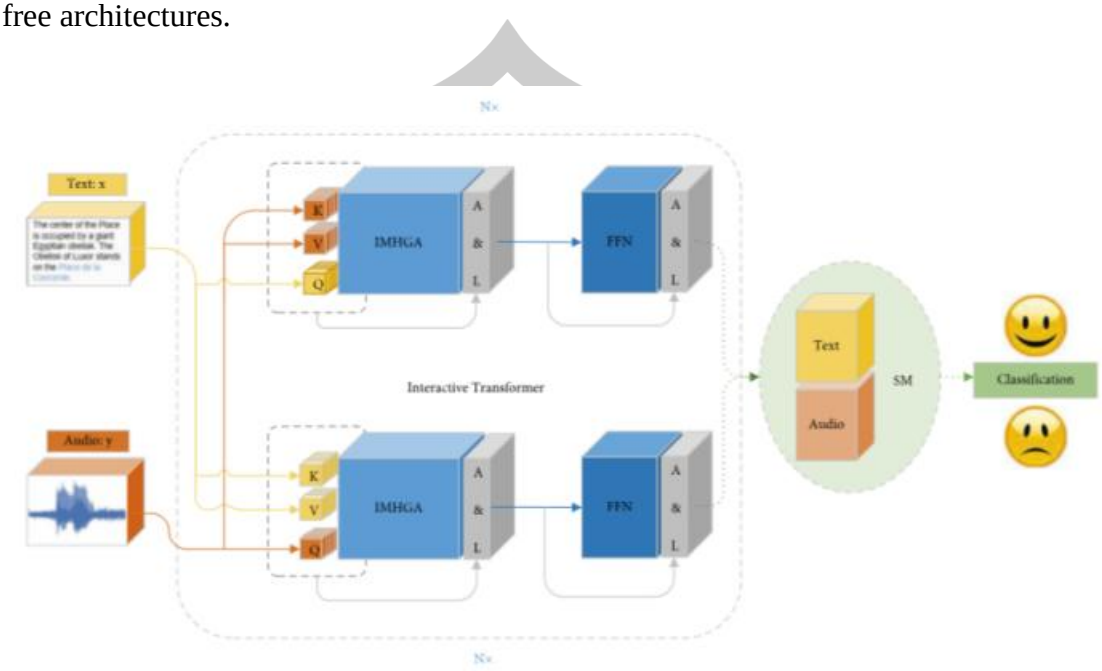



Figure 2.14 The Framework proposed by Zuhe Li et al [77]

From: [77]

Figure 2.15 illustrates the general framework of the model. The input consists of three modalities—text, video, and speech—while the output, denoted as y, represents emotion intensity. Initially, raw input data is transformed into numerical sequence vectors using a feature extractor for video and speech and a tagger for text. These vectors are then encoded into distinct unit-length representations, comprising two key components: feature fusion and maximum mutual information. The model operates in two modes: collaborative parts-fusion and mutual information (MI) maximization. In the fusion stage, the fusion network (FFF), composed of stacked linear activation layers, converts single-modal representations into a fused output (ZZZ), which is then used for final prediction through a regression-based Multilayer Perceptron (MLP). MI lower bounds are estimated and optimized at both the input and fusion levels. These components work together to generate backpropagated task-related and MI-related losses, enabling the model to infuse task-relevant information into the fusion process and enhance prediction accuracy.

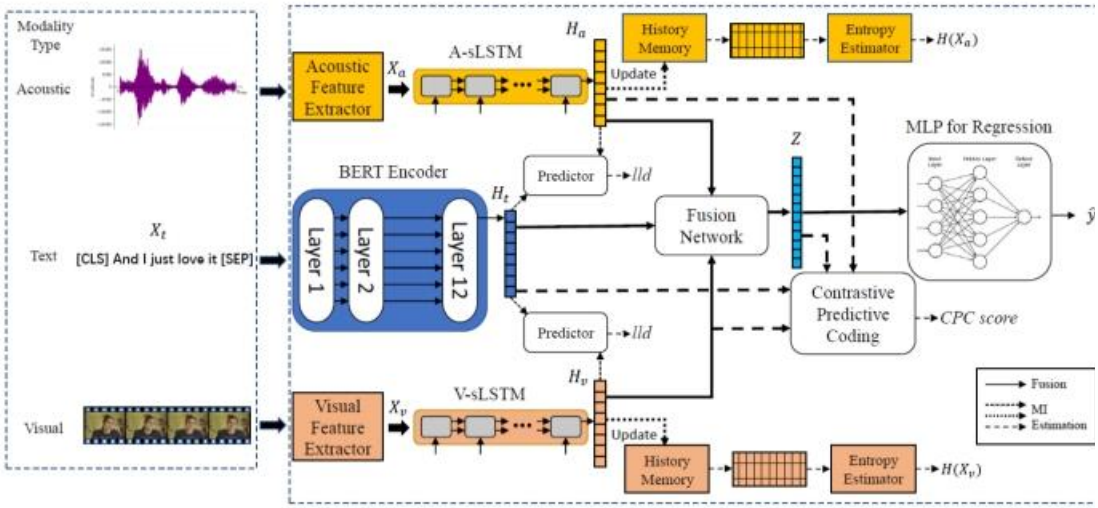



Figure 2.15 The overall architecture of the MMIM model

From: [78]

Experimental results on two widely used datasets demonstrate the effectiveness of this approach. While these methods produced promising results at the time, spatial interaction-based sentiment analysis models struggled to effectively capture semantic relationships both within and across modalities. Additionally, they relied heavily on manually engineered features, demanding significant human effort. Since the introduction of Google's Transformer model in 2017, numerous advancements in natural language processing (NLP) have been made. In particular, the Bidirectional Transformer Encoder Representations (BERT) model has achieved state-of-the-art performance across eleven NLP tasks, reinforcing its position as the leading model in the field.

In 2021, Radford et al. [79] introduced the Contrastive Language-Image Pretraining (CLIP) model, trained on a large-scale dataset of 400 million noisy images and text pairs collected from the internet. CLIP demonstrated remarkable effectiveness in representing multimodal information, enabling cross-modal interactions and delivering unexpected results in multimodal tasks. CLIP is designed to learn image representations through textual information. Its primary objective is to maximize the semantic similarity between positive image-text pairs in large datasets. This is achieved by leveraging both image and text encoders to extract multimodal embedded spatial information. The model aligns and compares textual content with visual representations, as illustrated in Figure 2.16. In the encoding process, each sentence is processed by a text encoder—either a ResNet or a Transformer—

producing n vectors, denoted as T_N. Similarly, each image is encoded into N vectors using an image encoder, designated as I_N. These representations are then arranged into a matrix, where diagonal elements correspond to positive samples with minimal loss (indicating high semantic alignment), while non-diagonal elements represent negative samples with higher loss. The diagonal of the matrix captures strong correlations between matching image-text pairs, while all other elements are treated as negative samples. By eliminating the dependency on predefined category labels, CLIP reframes the classification task as a categorical matching problem, achieving results comparable to fully supervised approaches. This model has demonstrated outstanding performance in both image and text processing tasks.

The processing involves multiple steps. First, an image is input into the Image Encoder to extract its features. Next, relevant labels—such as 'plane,' 'car,' or 'dog'—are assigned. A key step in this process is prompt engineering, where each label is transformed into a structured format, such as 'A photo of a plane.' These text prompts are then processed through a Text Encoder, generating corresponding text features. Following this, the cosine similarity between the extracted text features and image features is computed. Finally, the similarity scores are passed through a Softmax layer to produce a probability distribution, determining the most likely match (as illustrated in Figure 2.16).

Unlike traditional image classification models, CLIP does not rely on large-scale labeled image datasets for training. Instead, it pretrains on unlabeled image and text data using self-supervised contrastive learning. This approach enables the model to understand the semantic relationships between images and text, significantly enhancing its generalizability.

Following the introduction of the CLIP model, numerous downstream tasks based on CLIP emerged, yielding promising experimental results. Fang et al. proposed the CLIP2Video network, which leverages the CLIP model to capture spatial semantics, accurately detects motion in video frames using time-difference blocks, and redistributes video tags based on temporal block pairs, thereby enhancing multimodal correlations. Similarly, Luo et al. introduced the CLIP4Clip framework to seamlessly transfer CLIP’s knowledge for video-language retrieval. While these approaches have demonstrated positive outcomes, they also introduce challenges such as contextual information fragmentation and limitations in long-term contextual learning [79]. Building on CLIP's foundational principles, several models have been developed across different domains:

Segmentation: GroupViT, LSeg

Target Detection: VilD, GLIP v1/v2

Video Processing: VideoCLIP, CLIP4Clip, ActionCLIP

Image Generation: CLIPasso, VQGAN-CLIP, CLIP-draw

Speech Processing: AudioCLIP

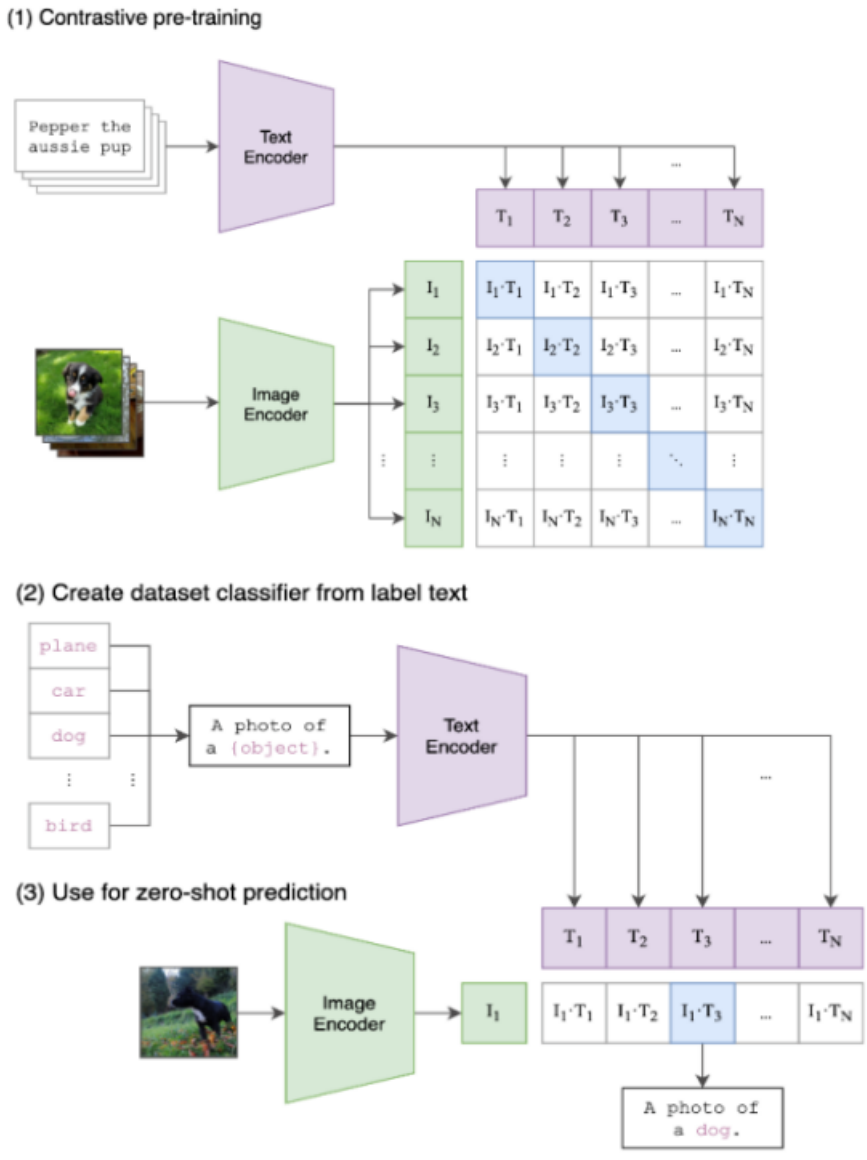



Figure 2.16 CLIP model

From: [79]

CLIP has ushered in a new era of multimodal representation learning, bridging computer vision (CV) and natural language processing (NLP). Remarkably, CLIP achieves results comparable to ResNet-50's supervised training on the ImageNet dataset—without requiring ImageNet data or labels—making it a zero-shot learning model. Due to its versatility, the CLIP pretraining approach has been successfully applied across a wide range of domains and applications.

Li and Weinberger [80] introduced a language-based approach to semantic segmentation, exploring the integration of CLIP into image segmentation models. Semantic segmentation, which can be considered a form of pixel-level classification, allows for the direct application of novel classification techniques and concepts. Language-driven Semantic Segmentation achieves zero-shot semantic segmentation in a manner similar to CLIP, where category prompts are provided as text input, and similarity is computed. The overall model structure closely resembles CLIP, as illustrated in Figure 2.17. The process begins with an image being passed through an image encoder (DPT ViT + decoder) to generate a feature vector, while textual input is processed through a text encoder to extract text feature vectors. These representations are then upscaled to match the original image size, followed by cross-entropy loss computation with ground truth labels. In semantic segmentation, pixel-dense features replace individual image-text features. To enhance this process, a new language-driven segmentation model, LSeg, was developed. LSeg employs a text encoder to encode descriptive input labels and an image encoder to compute pixel-by-pixel embeddings of images. The image encoder utilizes contrastive target training to align pixel embeddings with the corresponding text label embeddings. Since text embeddings offer a flexible label representation, the proposed model supports zero-shot inference. Additionally, traditional supervised segmentation models can be improved with text features. By combining text and image features through feature multiplication, the model learns language-aware features, enabling text prompts to generate diverse segmentation effects.

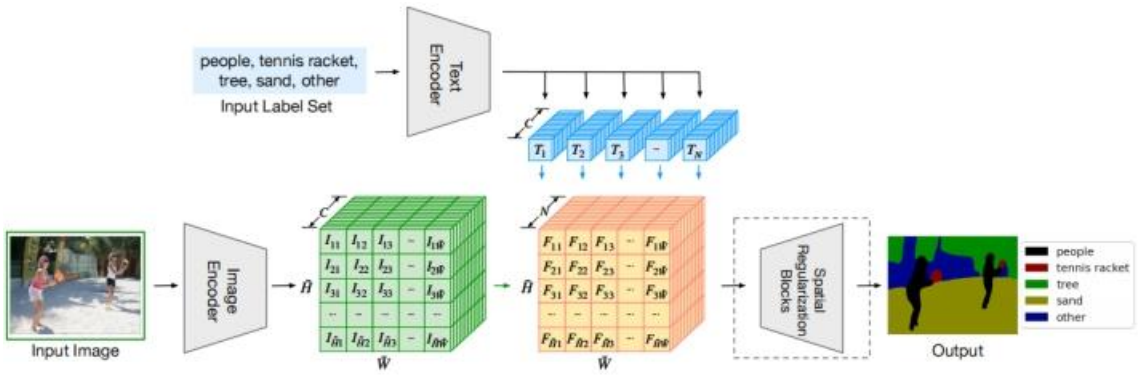



Figure 2.17 A framework of language-based semantic segmentation

From: [80]

The study results indicate that zero-shot learning outperforms few-shot and even one-shot learning. However, there is still significant room for further improvement. Although the approach described in this study is referred to as a text-driven model, it is not entirely unsupervised. The model was trained in a supervised manner, using seven segmented datasets along with predefined segmentation maps documented throughout the training process.

In [81], the study proposed incorporating the classical grouping process from segmentation into deep networks, enabling semantic segmentation to emerge automatically using only text supervision. This approach aimed to eliminate the need for manual labeling, leading to the development of GroupViT, a vision transformer-based model trained with text supervision for semantic segmentation. GroupViT introduced a Grouping Block into an existing Vision Transformer (ViT) model, along with learnable Group Tokens, as illustrated in Figure 2.18. The process begins by dividing an image into 196 small patches of size 16 × 16, similar to standard vision transformers. However, an additional 64 learnable patches are concatenated with the image patches before being passed through the transformer network. After six transformer layers, the grouping process is completed. The Group Block is then applied, where the concatenated patches—consisting of 196 × 384 image patches and 64 × 384 category patches—are transformed through the network. By leveraging transformer-based processing, the model blends confidence scores from different patches of the original image to generate distinct category patches, effectively achieving automatic grouping.

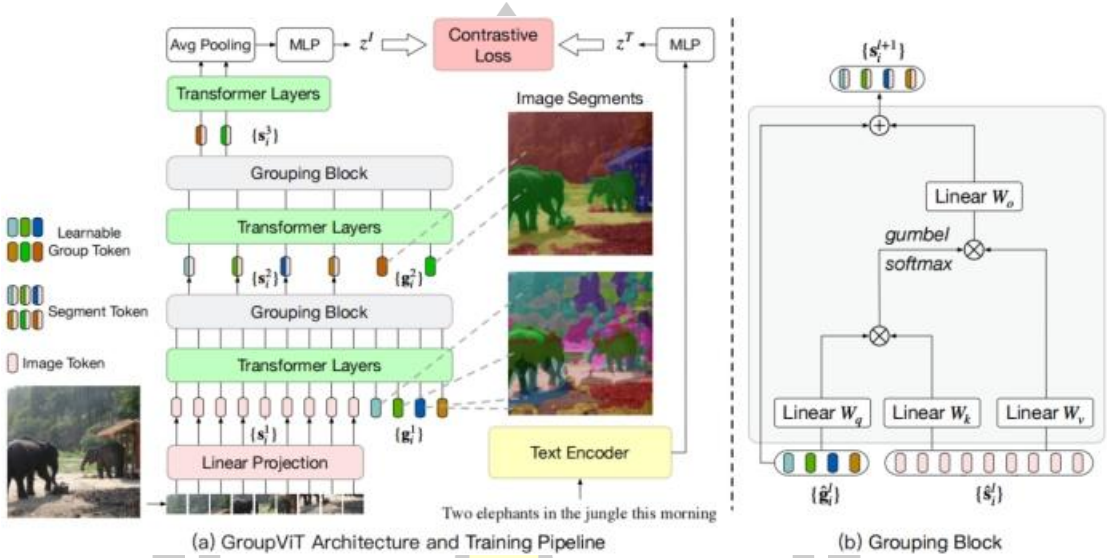



Figure 2.18 Adding a Grouping Block to an existing ViT model

From: [81]

The study took a crucial first step toward achieving zero-shot semantic segmentation through self-supervised learning, relying solely on text supervision without explicit human annotation. With GroupViT, representations learned from large-scale noisy image-text pairs could be effectively applied to semantic segmentation using a zero-shot approach. Additionally, the study highlighted that beyond image classification, text supervision could be extended to more fine-grained visual tasks—an area previously unexplored—paving the way for exciting new research opportunities.

In [82], Xiuye Gul et al. explored leveraging knowledge from a pre-trained open-world classification model to implement open-vocabulary object detection. Their study raised a key question: Is it possible to develop a more advanced target detector that can identify categories beyond those labeled in the training data? Or would expanding the model's vocabulary enable it to recognize a broader range of categories more effectively? Building on this idea, the authors aimed to train an open-vocabulary object detector capable of capturing a wider variety of target category labels in images.

As illustrated in Figure 2.19, in ViLD-Text, text embeddings are generated by inputting category names into a pre-trained text encoder. The inferred text embeddings are then used to classify detected regions in an image. The authors found

that encoding the similarity between visual concepts while training text embeddings with visual data yields better results than learning text embeddings solely from a textual corpus. Previous research achieved an average precision (AP) of 10.1 APr on the LVIS dataset—representing the model's performance on new categories—compared to only 3.0 APr using GloVe embeddings.

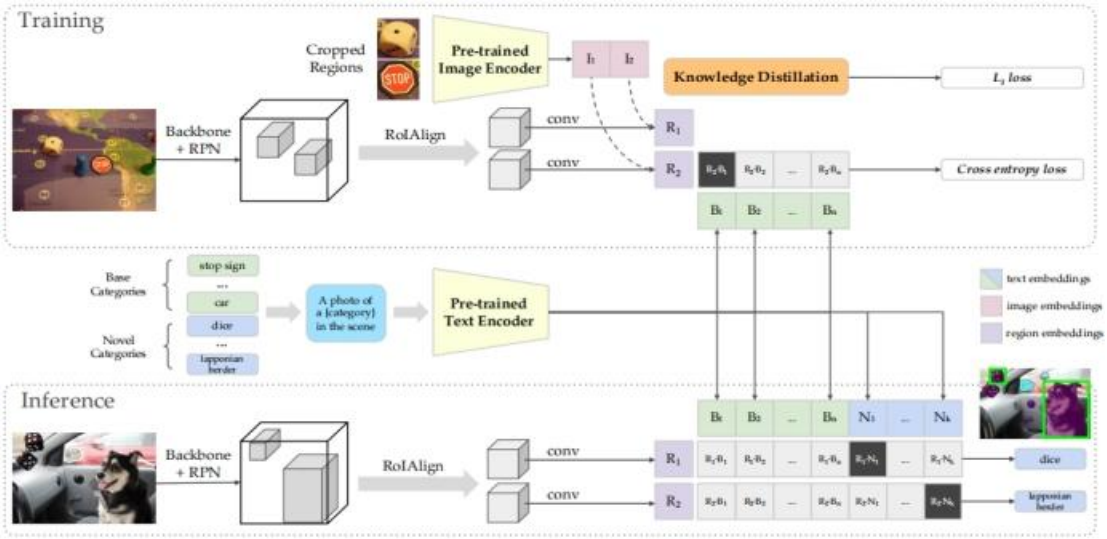



Figure 2.19 An overview of using ViLD for open-vocabulary object detection

From: [82]

In ViLD-Image, the same category information is input into a pre-trained image encoder to generate image embeddings. A MAScr-CNN model is then trained, aligning the detected bounding box areas with these embeddings. Unlike ViLD-Text, ViLD-Image extracts information from both base and novel categories. This is because the target proposal regions input to the image encoder may contain new category information, whereas ViLD-Text can only learn from predefined classification categories. The trained models used in the experiment included CLIP and ALIGN, as proposed by Jia et al. in 2021. Additionally, various model architectures, such as Vision Transformer (ViT) and other efficient networks, could be employed.

This method was designed to transfer knowledge from a pre-trained open-vocabulary image classification model (teacher) to a two-stage object detector (student). Specifically, the authors employed a learned model to encode image regions based on class text and object proposals. A student detector was then trained to identify bounding box embeddings that aligned with the text and image embeddings

generated by the teacher model. Using LVIS as a baseline, all rare categories were treated as novel and excluded from detection during training. The authors considered this approach a significant advancement toward open-vocabulary object detection. The robust CLIP-based pretraining model enables truly open-world object detection. Structurally, this method achieved an average precision (AP) of 27.6 for novel categories—an improvement of nearly five points over the previous Zareian approach—marking a significant leap in performance for this domain.

In [83], Li et al. introduced the concept of phrase grounding, suggesting that a model could learn to understand deeper relationships between images and sentences. Building on this idea, they proposed the Grounded Language-Image Pre-training (GLIP) model.

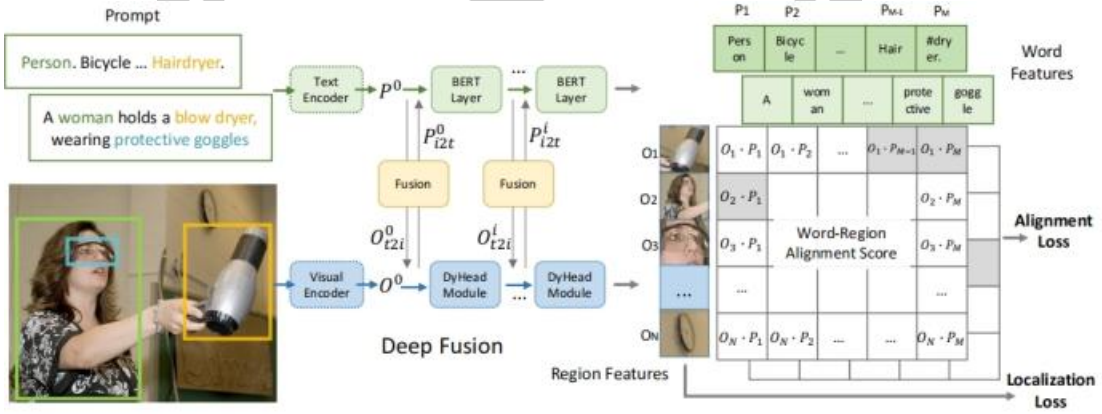



Figure 2.20 A unedified framework for detection and grounding

From: [83]

CLIP has bridged the gap between text and images, making it well-suited for classification tasks. However, GLIP extends this technology to more complex applications, such as object detection. As illustrated in Figure 2.20, traditional models like CLIP typically use image and text data only at the final stage to compute contrastive loss, a technique known as the late-fusion model. In contrast, GLIP introduces deep fusion by integrating image and text features more extensively, including fusion within the final layers of the encoder. GLIP employs DyHead to encode images and BERT to encode text. DyHead, a self-attention mechanism, operates across three dimensions: scale, space, and task, enhancing the model's ability to process multimodal information effectively.

Deep fusion enhanced phrase grounding performance by aligning image and text data, enabling text prompts to influence the detection model's predictions. GLIP leveraged large-scale pretraining data, making it easier for its pretrained models to adapt to downstream tasks. After fine-tuning on the COCO dataset, the pretrained GLIP achieved an average precision (AP) of 60.8 on the 2017 validation set and 61.5 on the test-dev set, surpassing existing state-of-the-art (SOTA) models. As a one-model-fits-all approach, GLIP demonstrated versatility across multiple tasks. Without using any additional annotations, it achieved 49.8 AP on the COCO val2017 dataset and 26.9 AP on the LVIS dataset.

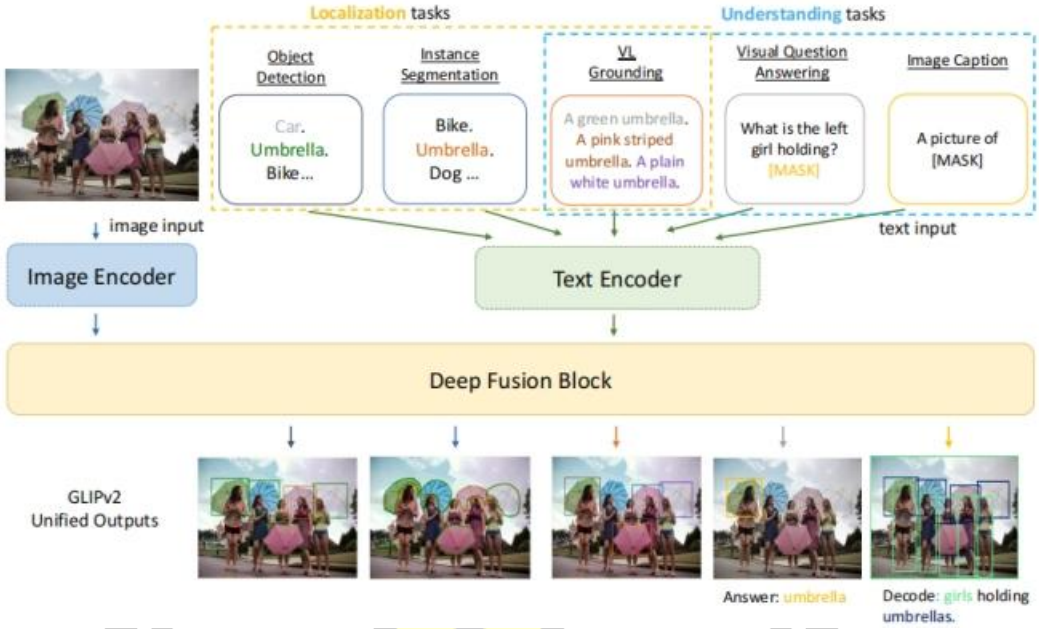



Figure 2.21 The framework of GLIPv2

From: [84]

In [84], Zhang et al. introduced GLIPv2, a grounded vision-language (VL) understanding model designed for both localization tasks (e.g., object detection, instance segmentation) and VL understanding tasks (e.g., visual question answering (VQA), image captioning). Image captioning, which has gained significant attention in recent years, requires models capable of effectively identifying and understanding objects. However, there are key differences between localization and VL understanding tasks. Localization is a vision-only task that demands fine-grained outputs (e.g., bounding boxes or pixel masks), whereas VL understanding focuses on multimodal fusion and requires high-level semantic outputs (e.g., answers or captions). Building on GLIP's approach of redefining object detection as a generalized phrase-

grounding task, GLIPv2 unifies localization and VL understanding into a single grounded vision-language framework. In this model, images and text are processed simultaneously, generating outputs for both object-level understanding (e.g., detection, segmentation) and image-level understanding (e.g., VQA, image captioning).

According to [85], the growing volume of videos uploaded online daily has led to a rising demand for efficient video-text retrieval, enabling users to quickly find relevant content. Beyond its practical applications on the web, video-text retrieval is also a significant research topic in multimodal vision and language understanding. To address this challenge, Huaishao et al. introduced CLIP4Clip, a video retrieval model that seamlessly transfers knowledge from the CLIP model to video-language retrieval in an end-to-end manner.

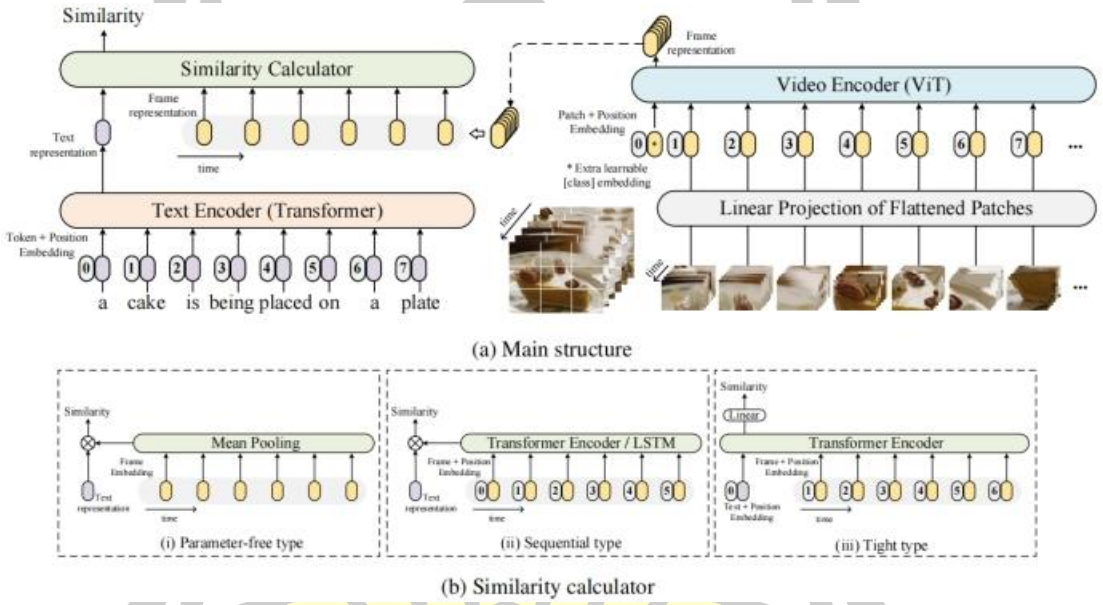



Figure 2.22 The framework of CLIP4Clip

From: [85]

The overall structure is illustrated in Figure 2.22. Both the Video Encoder and Text Encoder leverage the parameters of the CLIP model to process video and text data. The Video Encoder encodes each video frame individually, generating frame-by-frame representations, while the Text Encoder converts the video's textual description into a text representation. The similarity score between these representations is then computed. This study primarily contributed to the development of three similarity calculation methods based on the pretrained CLIP model: (1)

parameter-free, (2) sequential, and (3) constrained similarity calculators. These approaches were used to derive the final retrieval results. Additionally, CLIP was further trained on an extended vision-language dataset to enhance retrieval efficiency and improve the representation space for multimodal retrieval tasks.

The model's objective was to develop functions that compute the similarity between a given collection of texts and videos (or video clips). For text-to-video retrieval, the goal was to rank all videos (or video clips) based on their similarity to a given text query. Conversely, for video-to-text retrieval, the aim was to rank all text descriptions based on their relevance to a given video (or video clip). The model sought to assign high similarity scores to related video-text pairs while ensuring low scores for unrelated pairs. This study defined a video (or video clip) as a sequence of frames (images) composed of individual sampled frames. As a result, the proposed model adopted an end-to-end (E2E) approach, directly processing raw pixels by receiving frames as input for training.

In [86], the authors tackled the challenge of video-text matching within a multimodal learning framework. Their approach enhances video representation by incorporating richer semantic language supervision, enabling the model to achieve zero-shot action recognition without requiring additional annotated data or parameter constraints.

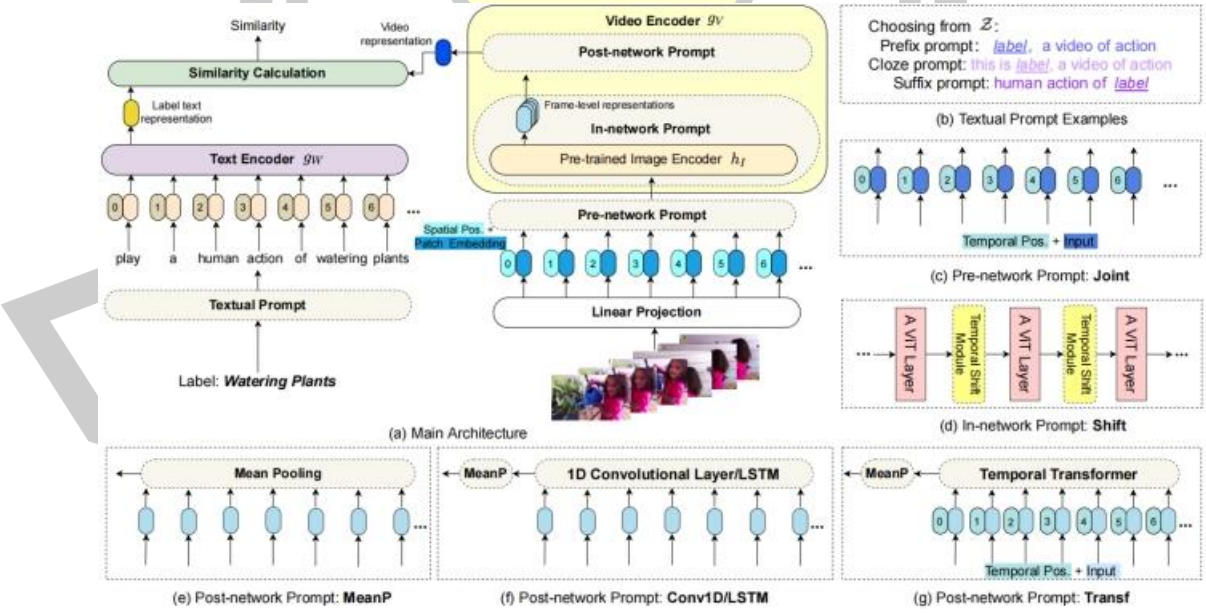



Figure 2.23 An Overview of ActionCLIP

From: [86]

The overall framework is illustrated in Figure 2.23. This study introduces a new paradigm for action recognition—pre-train, prompt, and fine-tune—which leverages large-scale internet data to pre-train multimodal models while addressing the challenges of costly and resource-intensive pretraining. Through prompting, the approach retains the strong representational capabilities of pretrained models. The paradigm follows three key steps:

Pre-training – Learning powerful representations from large-scale web image-text datasets.

Prompting – Adapting the action recognition task to resemble a pretrained problem using prompt engineering.

Fine-tuning – Optimizing the target dataset for peak performance.

As a result, the authors introduced ActionCLIP, a model that not only exhibits superior adaptability in zero-shot and few-shot scenarios but also achieves state-of-the-art performance in action recognition tasks. Using ViT-B/16 as the backbone, ActionCLIP attained a top-1 accuracy of 83.8% on the Kinetics-400 dataset. By redefining action recognition as a video-text multimodal learning challenge, this study provides a fresh perspective on action video analysis. The proposed pre-train, prompt, and fine-tune paradigm allows models to leverage powerful pretrained frameworks, significantly reducing the costs associated with traditional pretraining. ActionCLIP exemplifies this approach, demonstrating outstanding performance in everyday action recognition and zero-shot/few-shot scenarios.

In [87], Kim et al. introduced a simple vision-language pretraining (VLP) model called Visual and Language Transformer (ViLT). This model significantly simplifies the processing of visual data by employing a convolution-free approach, making it as efficient as text input processing. As illustrated in Figure 2.24, the model's general framework consists of three input modalities: text, video, and speech. The output, denoted as y, represents emotion strength. Initially, raw input is transformed into a numerical sequence vector using a feature extractor (for video and speech) and a tagger (for text). These representations are then encoded into a unified unit-length vector with two key components: feature fusion and maximum mutual information (MI). The model operates in two distinct modes:

Collaborative Parts-Fusion – Integrating multimodal representations for a comprehensive understanding.

MI Maximization – Enhancing feature learning by preserving relevant task-related information.

ViLT effectively streamlines multimodal data processing while maximizing information retention, making it a highly efficient approach for vision-language learning.

During the fusion stage, the single-modal representation is transformed into the fusion result (Z) through a stacked linear activation layer fusion network (FFF). This fused representation is then processed by a regression-based Multilayer Perceptron (MLP) to generate the final prediction.

The mutual information (MI) component estimates and optimizes the MI lower bound at two levels:

Input Level – Capturing essential modality-specific information.

Fusion Level – Enhancing cross-modal representation learning.

By jointly leveraging both components, the model generates backpropagated losses associated with the task and MI, allowing it to integrate task-specific information into the fusion outcome. This process improves the precision of the primary task prediction. To validate the effectiveness of the proposed method, experiments were conducted on two widely used datasets, demonstrating its efficacy.

ViLT is one of the simplest vision-and-language models to date, with the only comparable approach being the Transformer for multimodal fusion. Unlike previous models that relied on ResNet for feature extraction and additional networks for object detection, ViLT eliminates the need for these components. As a result, ViLT significantly reduces runtime and requires fewer parameters while maintaining performance. Notably, it achieves minimal or negligible performance degradation even without Region Features or CNN-based processing—an achievement that was previously unattainable in vision-language modeling.

This study, titled 'ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision,' introduces a minimal vision-language pretraining (VLP) model that eliminates the need for traditional embedding methods such as Faster R-CNN and ResNet for feature extraction. Instead, ViLT employs a

simple Patch Embedding approach to extract image features. To enhance model performance, various data augmentation techniques are applied during training, including whole-word masking in text processing and image augmentation in the visual component. ViLT proposes a faster, convolution-free method by dividing images into small patches, which are then transformed into embeddings using a linear projection layer. This replaces conventional, computationally intensive image feature extraction methods. The image is segmented into patches based on its attributes, followed by planarization. The resulting feature representations are then passed through a fully connected layer, serving as the input for the VLP model.

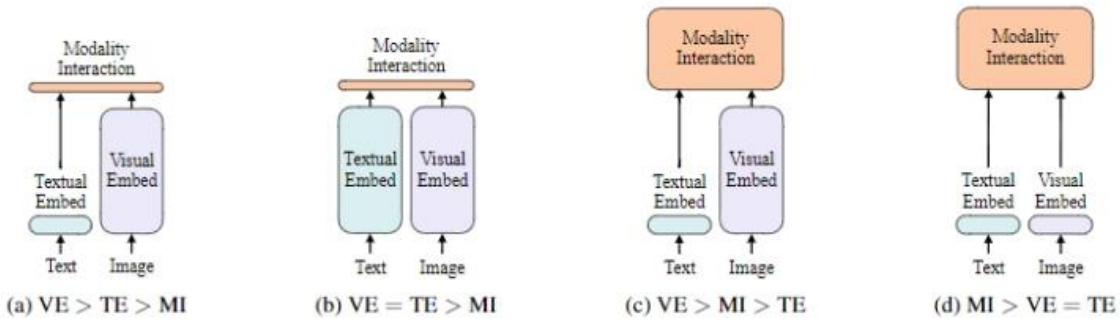



Figure 2.24 Four categories of vision-and-language models

From: [87]

The height of each rectangle in Figure 2.24 represents its proportional computational cost. The figure illustrates VE (Visual Embedder), TE (Textual Embedder), and MI (Modality Interaction). In ViLT, feature extraction is minimized, with the majority of computations shifted to modality fusion, resulting in faster inference speeds. One of ViLT's key innovations in multimodal learning is the removal of region features from the framework, significantly streamlining the model. Its major achievement lies in outperforming previous models in terms of both speed and efficiency. ViLT operates with remarkably fast processing times while maintaining competitive accuracy. Although ViLT does not achieve state-of-the-art (SOTA) performance, it introduces a novel supervision method that eliminates the need for convolutions or region-based features. While its accuracy is slightly lower than some prior models, ViLT offers substantial improvements in simplicity and processing speed, making it a highly efficient alternative in vision-language modeling.

In [88], the authors addressed key challenges in target detection, including high labeling costs, significant computational demands, and the inherent noise present

in large-scale image-text datasets collected from the web. Their proposed method enhances the understanding of visual and textual representations while eliminating the need for explicit image annotations, even when dealing with low-resolution images. To improve learning from noisy data, the study introduces a momentum distillation approach, a self-training method that leverages pseudo-targets generated by momentum models. Additionally, the authors provide a theoretical analysis of the proposed approach from the perspective of mutual information maximization. This analysis demonstrates how various training challenges can be reframed as different models for image-text description, offering new insights into multimodal learning.

The model architecture of ALBEF consists of three key components: an image encoder, a text encoder, and a multimodal encoder, as illustrated in Figure 2.25. For the image encoder, the model employs a 12-layer Vision Transformer (ViT-B/16), initialized with pre-trained weights from ImageNet-1k. The input image (I) is encoded into a sequence of feature tokens, which are aligned with an embedded [CLS] token.

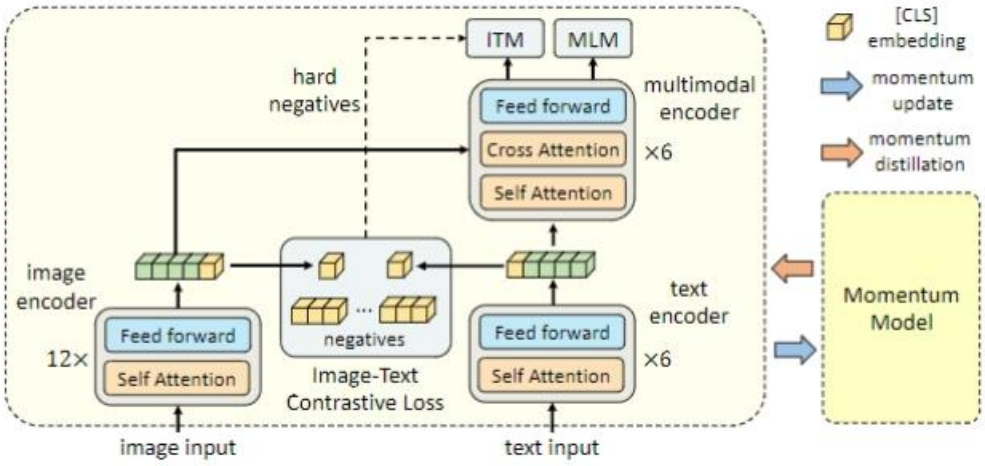



Figure 2.25 Illustration of ALBEF

From: [88]

The ALBEF model incorporates a 6-layer Transformer for both the text encoder and the multimodal encoder. The multimodal encoder is initialized using the final six layers of the BERT base model, while the text encoder is initialized with the first six layers. The input text (T) is first transformed into embedded text tokens by the text encoder, which then passes them to the multimodal encoder. Within the multimodal encoder, text and image features are integrated through cross-attention at every layer. The ALBEF pretraining phase consists of three key tasks:

1) Image-Text Contrastive Learning
2) Masked Language Modeling
3) Image-Text Matching

The overall pretraining objective is the sum of these three loss functions. ALBEF demonstrates state-of-the-art performance across various vision-language tasks. Notably, when trained on larger datasets, ALBEF surpasses previous pretraining approaches in image-text retrieval. On VQA and NLVR2, ALBEF outperformed the most advanced existing models by absolute margins of 2.37% and 3.84%, respectively, while also achieving faster inference speeds

[89] introduced CapFilt, a data augmentation approach designed for large-scale datasets. The researchers observed that the large-scale data retrieved by CLIP from the internet was of relatively low quality, which negatively impacted model performance. To address this, the study incorporated three types of loss functions during pretraining:

1) ITC (Image-Text Contrastive Learning) – Optimizes image and text alignment.
2) ITM (Image-Text Matching) – Enhances the model's ability to associate images with corresponding text.
3) LM (Language Modeling) – Improves the generation of textual descriptions.

By applying CapFilt, the study aimed to refine data quality and enhance the effectiveness of multimodal learning on large-scale datasets.

CapFilt primarily consists of two components: Captioner and Filter, as illustrated in Figure 2.26. The Captioner generates image titles, while the Filter removes low-quality text pairs. Both components are derived from a pre-trained multimodal model that has been fine-tuned on the COCO dataset, which contains higher-quality data. The data enhancement process occurs in two stages:

1) Fine-tuning Stage – Manually labeled high-quality data is used to fine-tune both the Captioner and the Filter.
2) Filtering Stage – The Filter (essentially an encoder) is applied to scan and refine large-scale web data.

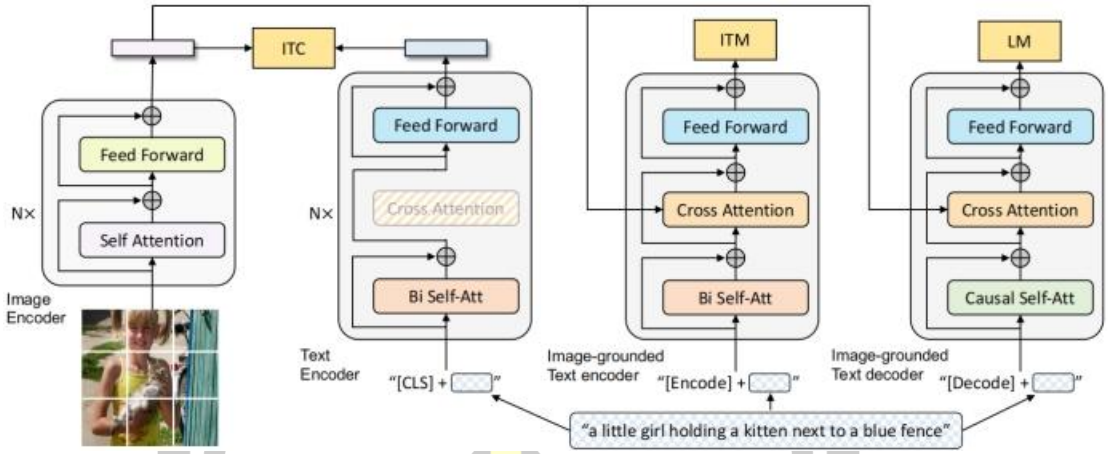
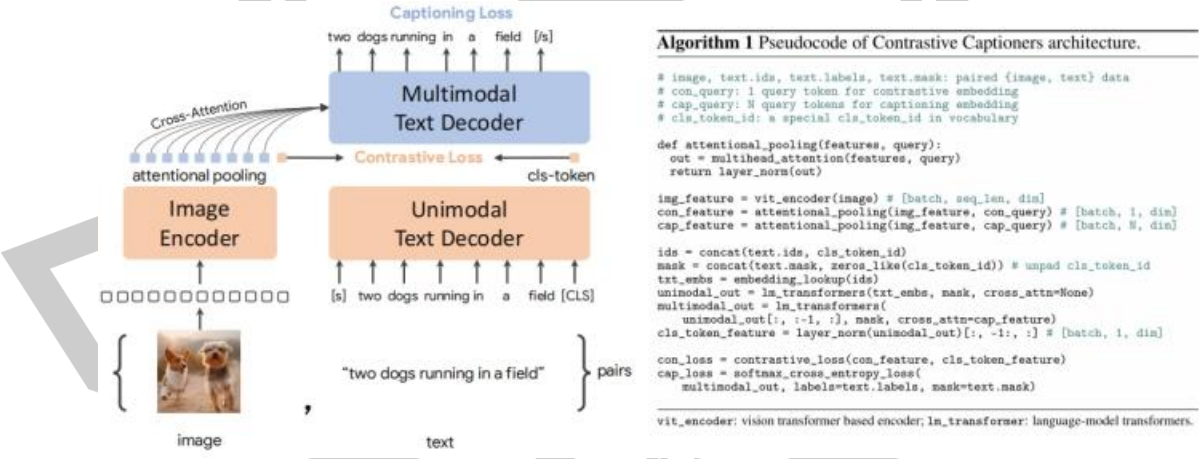



Figure 2.26 A CapFilt approach

From: [89]

In this process, the ITC (Image-Text Contrastive) and ITM (Image-Text Matching) loss functions are computed and used to eliminate text pairs with weak correspondence. The Captioner (primarily a decoder) then generates new image titles based on the refined network images. Finally, the text produced by the Captioner replaces the low-quality text filtered out by the Filter. The cleaned and enriched dataset is then combined and fed into the model for training, ensuring higher-quality multimodal learning.




**Algorithm 1** Pseudocode of Contrastive Captioners architecture.

```
# image, text.ids, text.labels, text.mask: paired {image, text} data
# con_query: 1 query token for contrastive embedding
# cap_query: N query tokens for captioning embedding
# cls_token_id: a special cls_token_id in vocabulary

def attentional_pooling(features, query):
  out = multihead_attention(features, query)
  return layer_norm(out)

img_feature = vit_encoder(image) # [batch, seq_len, dim]
con_feature = attentional_pooling(img_feature, con_query) # [batch, 1, dim]
cap_feature = attentional_pooling(img_feature, cap_query) # [batch, N, dim]

ids = concat(text.ids, cls_token_id)
mask = concat(text.mask, zeros_like(cls_token_id)) # unpad cls_token_id
txt_embs = embedding_lookup(ids)
unimodal_out = lm_transformers(txt_embs, mask, cross_attn=None)
multimodal_out = lm_transformers(
    unimodal_out[:, :-1, :], mask, cross_attn=cap_feature)
cls_token_feature = layer_norm(unimodal_out)[:, -1:, :] # [batch, 1, dim]

con_loss = contrastive_loss(con_feature, cls_token_feature)
cap_loss = softmax_cross_entropy_loss(
    multimodal_out, labels=text.labels, mask=text.mask)
```

vit_encoder: vision transformer based encoder; lm_transformer: language-model transformers.

Figure 2.27 Detailed illustration of CoCa architecture and training objectives

From: [90]

CoCa is a successor to ALBEF, with a model architecture that closely resembles ALBEF's structure, as illustrated in Figure 2.27. The left side of the figure represents the image encoder, while the right side represents the text decoder. In CoCa, an image token is first generated using the [CLS] token and the text token. The remaining image tokens are then created based on the initial image token. The text decoder, operating in a multimodal state, employs attention mechanisms to integrate visual and linguistic elements from both the image and text. Finally, the model applies the caption loss function to refine the learning process and improve multimodal understanding.

Table 2.3 Image classification and video action recognition with frozen encoder or fine-tuned encoder

From: [90]

| **Model** | **ImageNet** | **Model** | **K-400** | **K-600** | **K-700** | **Moment-in-Time** |
|---|---|---|---|---|---|---|
| **ALIGN** | 88.6 | ViViT | 84.8 | 84.3 | - | 38.0 |
| **Florence** | 90.1 | MoViNet | 81.5 | 84.8 | 79.4 | 40.2 |
| **MetaPseudo Labels** | 90.2 | VATT | 82.1 | 83.6 | - | 41.1 |
| **CoAtNet** | 90.9 | Florence | 86.8 | 88.0 | - | - |
| **ViT-G** | 90.5 | MaskFeat | 87.0 | 88.3 | 80.4 | |
| **ViT-G+Model Soups** | 90.9 | CoVeR | 87.2 | 87.9 | 78.5 | 46.1 |
| **CoCa(frozen)** | 90.6 | CoCa(frozen) | 88.0 | 88.5 | 81.1 | 47.4 |
| **CoCa(finetuned)** | 91.0 | CoCa(fine tuned) | 88.9 | 89.4 | 82.7 | 49.0 |

The key difference from ALBEF is that CoCa incorporates trainable image attention pooling, which enables the model to learn improved features tailored to specific tasks. The Decoder is used for both single-text and multimodal inputs, enhancing the model's versatility. By employing Captioning Loss and optimizing the Decoder structure, the model aims to improve operational efficiency and multimodal performance.

This study introduces Contrastive Captioner (CoCa), a minimalist design that combines the fundamental image-text encoder-decoder paradigm with contrastive loss and captioning loss for pre-training. CoCa achieves an impressive zero-shot image classification accuracy of 86.3% on ImageNet, with further improvements in zero-shot cross-modal retrieval results on the MSCOCO and Flickr30k datasets. When using the frozen encoder, CoCa attains 90.6% classification accuracy on ImageNet, and 88.0%, 88.5%, and 81.1% on the Kinetics-400, Kinetics-600, and Kinetics-700

datasets, respectively. It also achieves 47.4% on the Moments-in-Time dataset. After fine-tuning, CoCa's performance improves further, reaching 91.0% classification accuracy on ImageNet, 82.3% on VQA, and 120.6% on NoCaps. These results, as shown in Table 2.3, are quite remarkable.

In [91], Convolutional Neural Networks (CNNs) were employed for sentiment analysis in multimedia data. CNNs have been widely applied in multi-modal sentiment analysis research to extract features from different modalities. Two separate CNNs were used to independently extract features from each modality—image and text. These extracted features were then integrated into a CNN framework, which established the relationship between the two modalities, enabling sentiment polarity prediction. The datasets used for image-text multimodal sentiment analysis typically consist of blog posts written by individuals. These posts often contain significant implicit information and lack explicit sentiment words. As a result, emotions are not limited to simple happy or negative states but encompass a wide range of nuanced and concealed feelings.

In [58], the authors introduced a technique for multimodal sentiment analysis aimed at studying the organization of emotion. A novel multimodal feature extraction approach was developed by integrating findings from psychology and emotion research. This model was capable of extracting implicit information from the input and autonomously generating a logical vocabulary list.

In [92], the study introduced a technique for context-aware sentiment analysis of social media users. The researchers used a probabilistic model called CASA to analyze the sentiment of tweets. CASA integrates sentiment information using Bernoulli parameters and considers both the contextual semantic relationships and the semantic correlation between modalities. This approach has the potential to accurately predict emotions and produce favorable results. Previous methods overlooked implicit information, such as emoticons, in blog posts, leading to inaccurate sentiment analysis.

In [93], the study introduced a novel approach for transferring parameters and fine-tuning a model called TFCNN, designed for image sentiment categorization. The model aims to capture the underlying emotional meaning in images. Sentiment analysis plays a critical role in stock prediction and commodity recommendations by supporting decision-making. As a result, leveraging large-scale data is essential for accurately extracting genuine emotional insights from consumers.

In [94], a novel approach was introduced for integrating visual-textual sentiment analysis of social multimedia, utilizing a cross-modality consistent regression model (CCR). The dataset used in this study combined both machine-generated weak labels and manual labels. The findings highlighted the effectiveness and practicality of the approach.

In [95], the authors introduced a study on visual-textual sentiment analysis, focusing on the intersection of attention mechanisms and tree-structured recursive neural networks. The Cross-Modal Consistent Regression Model (CCR) processes modal information in a structured way, enabling effective data alignment across different modalities. The model employs a Long Short-Term Memory (LSTM) network with an attention mechanism to train the semantic relationships between text and image data jointly. This approach successfully incorporates contextual information and semantic correlation, leading to accurate sentiment orientation determination. The use of the attention mechanism played a key role in improving performance in multimodal sentiment analysis. In [96], the authors proposed a multi-level attention-based fusion mechanism for contextual multimodal sentiment analysis. This recurrent model considers contextual semantic information while measuring the relative importance of each modality. The model automatically generates influence scores for the input modal features, adjusting based on their importance

In [97], the authors introduced a multimodal sentiment analysis technique that incorporates word-level fusion and reinforcement learning. The proposed method utilizes a gated recurrent unit (GRU) architecture with a time attention mechanism for processing multimodal sentiment data. Word-level modal fusion is employed to tackle challenges posed by noisy data and the complexity of fusion in the dataset, enhancing the model's ability to effectively analyze sentiment across different modalities.

In [98], the authors introduced a technique for analyzing the sentiment of image-text pairs using a deep multimodal attentive fusion approach. They proposed a sophisticated model called Deep Multimodal Attention Fusion (DMAF). Initially, an attention network model was used to extract features from different modalities. Then, an attention mechanism that incorporates fusion technology was applied to merge the semantic connections between the image and text modalities. Finally, the three attention models were integrated to predict sentiment polarity.

In [99] This paper explores the incorporation of emojis into sentiment analysis (SA) to improve accuracy in Chinese text analysis. Due to the diversity and

variability of Chinese syntax and semantics, accurately identifying and distinguishing individual emotions from online texts is challenging. To address this limitation, the authors introduce emojis as a new source of sentiment, evaluating their impact on SA algorithms through comparisons of rule-based and classification approaches. The study finds that emojis are effective as expanding features for improving SA accuracy, and algorithm performance can be further enhanced by considering different emoji usages. Based on these findings, the authors propose an improved emoji-embedding model (CEmo-LSTM) based on Bi-LSTM, achieving the highest accuracy (around 0.95) when analyzing Chinese online texts. Finally, the algorithm is applied to a large dataset collected from Weibo during the COVID-19 pandemic (December 1, 2019, to March 20, 2020), revealing that the pandemic significantly impacted individual sentiments, leading to more passive emotions (e.g., fear and sadness).

As highlighted in previous studies, sentiment classification still faces several challenges. While there is a growing trend toward training large models, these models are often impractical for smaller tasks due to insufficient data and computational resources. A critical aspect of sentiment analysis for product reviews is selecting a high-quality pre-trained model, integrating adaptable modules, and tailoring the model to the specific product review dataset. The goal is to discover an effective sentiment analysis approach that improves both the accuracy and quality of sentiment predictions. The research in the aforementioned studies shows that pre-trained models such as CLIP, ALBEF, BILP, COCA, and BILP2 effectively address challenges like costly manual annotation, imprecise labeling, slow video feature extraction, and limited dataset sizes. With the rapid advancements in deep learning, pre-training technology has gained significant traction in the field of natural language processing (NLP). Researchers in other domains have adopted these methods to implement transfer learning, significantly improving data classification tasks. This work aims to mitigate subjective errors from manual screening and the cumulative errors from different NLP systems by proposing an emotional analysis pre-training technique and model based on traditional approaches.

# CHAPTER 3 RESEARCH METHODOLOGY

This chapter aims to present a detailed and comprehensive explanation of the proposed methodology for this study.

## 3.1 Datasets

The datasets utilized in this study were collected from the *Douban Film and Television Network website* (https://movie.douban.com/). They consist of Chinese-language movie reviews that incorporate both text and emoticons.

The first dataset comprises reviews related to the action film, *Chosin Lake's Watergate Bridge*, containing 400 reviews with 5-star ratings. The minimum, maximum, and average word counts in these reviews are denoted as 4, 633, and 42, respectively.

The second dataset consists of movie reviews from various films, totaling 1,799 reviews with 5-star ratings. The minimum, maximum, and average word counts in these reviews are recorded as 2,464, and 71, respectively.

Using two datasets helps to comprehensively evaluate the performance and generalization ability of the model, ensuring its effectiveness in different application scenarios. The specific reasons are as follows:

**Data Diversity**: The first dataset focuses on reviews of a specific action movie ("Chosin Lake's Watergate Bridge"), while the second dataset covers reviews of multiple movies. This diversity helps verify the model's generalization ability across different types of movie reviews.

**Data Scale**: The first dataset contains 400 reviews, while the second dataset contains 1799 reviews. A larger dataset can help the model better learn complex patterns and reduce the risk of overfitting.

**Sentiment Distribution**: The sentiment distribution in the two datasets is different. The first dataset has a relatively balanced number of positive, neutral, and negative reviews, while the second dataset is dominated by positive reviews. This difference helps evaluate the model's performance under different sentiment distributions.

**Application Scenarios**: By using two datasets, different real-world application scenarios can be simulated, such as sentiment analysis of a single movie and comprehensive sentiment analysis of multiple movies, thereby validating the model's applicability.

Both datasets were employed in two experimental setups: binary sentiment classification and multiclass sentiment classification (three classes). Movie reviews in both datasets were categorized into three sentiment classes based on their star ratings. Reviews with 1-star and 2-star ratings were assigned as negative, indicating unfavorable opinions. Reviews with a 3-star rating were classified as neutral, reflecting a balanced or mixed sentiment. Meanwhile, reviews with 4-star and 5-star ratings were assigned as positive, representing favorable feedback. This classification

framework helps capture varying degrees of sentiment, enabling a more comprehensive sentiment analysis.

These datasets were manually collected to ensure the inclusion of multimodal data, specifically reviews containing both text and emoticons. An overview of the movie review collection process is presented in Figure 3.1.

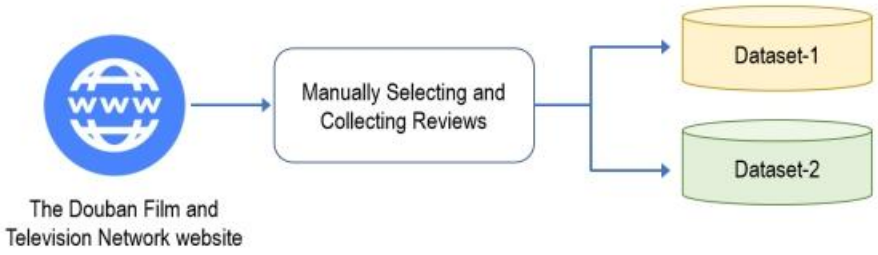



Figure 3.1 An overview of the movie review collection process

The two datasets contain a total of 60 unique emoticons. These emoticons, embedded within the reviews, contribute to the multimodal nature of the data by conveying emotions and enhancing textual expressions. Their presence adds an additional layer of sentiment representation, making them valuable for sentiment analysis. By incorporating both textual content and emoticons, the datasets provide a richer and more nuanced understanding of user opinions.

A summary of the dataset collected from the Douban Film and Television Network website is provided in Table 3.1, while Table 3.2 presents an overview of the datasets utilized in this study.

Table 3.1 Summary of datasets collected from the Douban Film and Television Network website

| Datasets | Number of movie reviews in each rating label | | | | | Total number of movie reviews |
|---|---|---|---|---|---|---|
| | 1-star | 2-star | 3-star | 4-star | 5-star | |
| 1 | 27 | 68 | 124 | 106 | 75 | 400 |
| 2 | 22 | 52 | 166 | 447 | 1,112 | 1,799 |

Table 3.2 An overview of the datasets utilized in this study

| Datasets | Number of movie reviews in positive class label | Number of movie reviews in neutral class label | Number of movie reviews in negative class label |
|---|---|---|---|
| 1 | 181 | 124 | 95 |
| 2 | 1,559 | 166 | 74 |

Some examples of movie reviews related to the action film Chosin Lake's Watergate Bridge are presented in Table 3.3.

Table 3.3 Examples of movie reviews related the action movie, Chosin Lake's Watergate Bridge

| Ratings | Original reviews | Chinese-to-English Translation |
|---|---|---|
| Positive | 《长津湖之水门桥》真是太震撼了！战斗场面逼真，演员演技精湛，情节紧凑。强烈推荐！👍🔥 | Chosin Lake's Watergate Bridge is truly breathtaking! The battle scenes are realistic, the actors deliver outstanding performances, and the plot is intense. Highly recommended! 👍🔥 |
| Neutral | 《长津湖之水门桥》的续集还算不错。战斗场面制作精良，但剧情有些可预测。整体来说，这是一部中规中矩的动作片。😐🎥 | The sequel to Chosin Lake's Watergate Bridge was decent. The battle scenes were well-executed, but the storyline felt a bit predictable. Overall, it's an average action film. 😐🎥 |
| Negative | 我对《长津湖之水门桥》感到失望。剧情支离破碎，人物塑造欠缺。动作场面无法弥补薄弱的故事情节。不推荐。😞👎 | I was disappointed with Chosin Lake's Watergate Bridge. The plot was disjointed, and the character development was lacking. The action scenes couldn't compensate for the weak storyline. Not recommended. 😞👎 |

## 3.2 Tools used in the study

Several Python libraries are widely used for sentiment classification, covering tasks such as data processing, natural language processing (NLP), feature extraction, and machine learning modeling. Table 3.4 presents some of the key libraries and tools.

Table 3.4 Summary of Python libraries used in this study

| Libraries | Objectives of usage |
|---|---|
| Pandas | A powerful library for data manipulation and analysis, particularly useful for handling structured data such as CSV files. |
| NLTK (Natural Language Toolkit) | A comprehensive NLP library that provides tools for tokenization, stemming, lemmatization, and stop-word removal, essential for text preprocessing. |
| scikit-learn | A versatile machine learning library offering tools for data preprocessing, feature extraction, model selection, and evaluation. It includes implementations of various classification algorithms, such as Naïve Bayes, Support Vector Machine (SVM), and Random Forest. |
| TensorFlow & PyTorch | Deep learning frameworks that provide scalability and flexibility for building and training neural network models, especially for large-scale sentiment classification tasks. |
| Keras | A high-level neural network API running on top of TensorFlow or Theano, offering an intuitive interface for designing and training deep learning models. |
| Gensim | A library designed for topic modeling, document indexing, and similarity retrieval across large text corpora. It is particularly useful for word embeddings and topic modeling in NLP tasks. |
| TextBlob | A simple yet effective library for processing textual data, including sentiment analysis, part-of-speech tagging, and noun phrase extraction. |
| Word2Vec, GloVe, and FastText | Pre-trained word embedding models that capture semantic relationships between words, improving feature representation for sentiment classification models. |
| Transformers (Hugging Face) | A powerful library for working with BERT and other transformer-based models. It allows for fine-tuning pre-trained BERT models for sentiment classification with minimal effort. |

## 3.3 An Overview of Research Methodology

Identifying sentiment in consumer reviews that contain both text and emoticons requires a multimodal sentiment analysis approach. The step-by-step process for this task is outlined in Figure 3.1.

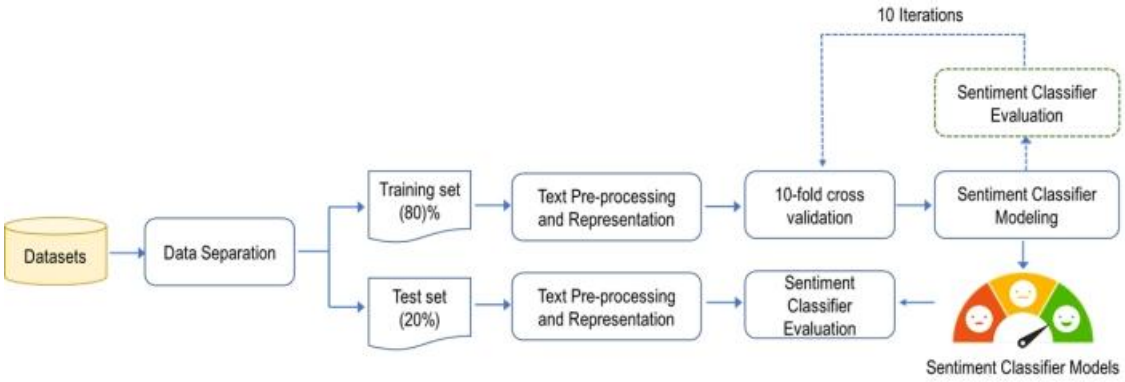



Figure 3.2 An overview of research methodology

The research methodology can be outlined step by step as follows.

### 3.3.1 Data Separation

The holdout method was used for data separation, where 80% of the data was allocated for training and 20% for testing. This approach ensures that the model is trained on a substantial portion of the dataset while maintaining a separate test set to evaluate its performance on unseen data.

### 3.3.2 Text Pre-processing and Representation

3.3.2.1 Text Pre-processing

Before training the sentiment classification model, the raw text data undergoes a pre-processing phase to enhance quality, reduce noise, and improve model performance. This process begins with tokenization, where the text is split into individual words or subwords to facilitate analysis. To maintain consistency, all text is converted to lowercase, while common stop words such as "*the*," "*is*," and "*and*" are removed to eliminate irrelevant information. Punctuation marks and special characters are also discarded to prevent unnecessary noise in the model.

3.3.2.2 Text Representation

In this study, multiple techniques are used to represent text, as it is essential to compare their performance. The methods employed include Word2Vec, GloVe, FastText, Ada-002, and BERT embedding, each offering unique advantages in capturing semantic relationships and contextual meanings. By evaluating these techniques, the study aims to determine the most effective representation for sentiment classification tasks.

***Word2Vec Representation*** **-** After pre-processing, the words are transformed into numerical representations using Word2Vec, a powerful word embedding technique that captures semantic relationships between words. Instead of representing words as discrete symbols, Word2Vec learns vectorized representations based on contextual associations, allowing words with similar meanings to have closer representations in vector space. The model is trained on the dataset to generate dense word embeddings, which are then used to map each word to a multi-dimensional vector. Unlike traditional approaches such as one-hot encoding, Word2Vec preserves the contextual meaning of words, enabling the model to recognize relationships between words and improve sentiment classification performance. By integrating effective text pre-processing with Word2Vec-based word representations, the dataset is structured in a way that enhances learning, ultimately leading to more accurate sentiment classification results.

Word2Vec generates word embeddings based on contextual relationships learned from a corpus. Each word in the sentence is mapped to a pre-trained word vector. An example of the Word2Vec representation for the sentence “I love this movie.” can be illustrated in Table 3.5.

Table 3.5 An example of the Word2Vec representation

| Words | Word2Vec Embedding (Example 300-D) |
|---|---|
| I | [0.21, -0.18, 0.67, ..., 0.09] |
| love | [0.45, 0.78, -0.31, ..., 0.52] |
| this | [-0.22, 0.36, -0.41, ..., -0.15] |
| movie | [0.61, -0.12, 0.85, ..., 0.33] |

This table presents the vectorized form of each word in the sentence, where each word is mapped to a high-dimensional numerical representation based on its contextual meaning within a trained Word2Vec model. These word embeddings capture semantic relationships between words, enabling more effective processing and analysis of textual data in various natural language processing tasks.

***GloVe Represention*** **-** Following pre-processing, the cleaned text is converted into numerical representations using GloVe (Global Vectors for Word Representation), an advanced word embedding technique that captures semantic meaning and contextual relationships between words. Unlike traditional methods such as one-hot encoding, which treats words as isolated entities, GloVe leverages co-occurrence probabilities to learn how words are related based on their appearances in a large corpus. This technique constructs a dense vector space where words with similar meanings have closer representations. The word vectors generated by GloVe enable the model to understand complex linguistic patterns, improving its ability to differentiate sentiment nuances in text. By integrating text pre-processing with GloVe-based word representations, the dataset is transformed into a structured format, allowing the sentiment classification model to achieve higher accuracy and better generalization.

GloVe constructs word embeddings based on word co-occurrence statistics. Each word is mapped to a vector from a pre-trained GloVe model. An example of the GloVe representation for the sentence “I love this movie.” can be illustrated in Table 3.6.

Table 3.6 An example of the GloVe representation

| Words | GloVe Embedding (Example 300-D) |
|---|---|
| I | [0.15, -0.12, 0.59, ..., 0.08] |
| love | [0.50, 0.71, -0.28, ..., 0.49] |
| this | [-0.19, 0.40, -0.35, ..., -0.10] |
| movie | [0.55, -0.18, 0.80, ..., 0.29] |

This table showcases the numerical vector representation of each word, generated using the GloVe model, which captures both global and local statistical information from a large corpus. By leveraging co-occurrence probabilities, GloVe embeddings effectively encode semantic relationships between words, enhancing various natural language processing tasks by providing meaningful and contextually rich word representations.

***FastText Representation*** - Once pre-processing is complete, the cleaned text is converted into word embeddings using FastText, a powerful word representation technique that extends traditional word embeddings by incorporating subword information. Unlike models like Word2Vec, which treat words as single entities, FastText breaks words down into smaller character n-grams. This approach allows the model to understand words based on their morphological structure, making it particularly effective for handling misspellings, rare words, and out-of-vocabulary terms. The advantage of FastText embeddings lies in their ability to capture not only the contextual meaning of words but also linguistic variations, making them more robust for sentiment classification tasks. By integrating text pre-processing with FastText-based word representations, the dataset is transformed into a structured numerical format that enhances the model's ability to learn sentiment patterns effectively, ultimately leading to improved classification accuracy and better generalization across diverse text inputs.

FastText improves upon Word2Vec by representing words as subword n-grams, making it effective for handling misspellings and rare words. Each word is mapped to a FastText embedding. An example of the FastText representation for the sentence “I love this movie.” can be illustrated in Table 3.7.

Table 3.7 An example of the FastText representation

| Words | FastText Embedding (Example 300-D) |
|---|---|
| I | [0.18, -0.16, 0.65, ..., 0.11] |
| love | [0.48, 0.75, -0.30, ..., 0.51] |
| this | [-0.20, 0.38, -0.39, ..., -0.13] |
| movie | [0.60, -0.15, 0.83, ..., 0.31] |

***Ada-002 Representation*** - After pre-processing, the text data is converted into numerical representations using Ada-002, an advanced OpenAI embedding model designed for natural language processing (NLP) tasks, including text classification, clustering, and semantic search. Unlike traditional word embedding techniques such as Word2Vec or FastText, Ada-002 embeddings leverage transformer-based deep learning architectures, allowing them to capture complex linguistic relationships and contextual nuances in the text. The primary advantage of using Ada-002 embeddings is their ability to generate high-quality semantic representations of words, phrases, and sentences. These embeddings understand

contextual meanings, relationships between words, and even subtle sentiment shifts, making them highly effective for sentiment classification tasks. By integrating text pre-processing with Ada-002-based representations, the dataset is transformed into a structured format that enhances the model's ability to detect sentiment patterns accurately, leading to improved classification performance and robustness across different types of text inputs.

Ada-002 (OpenAI Embedding Model) provides a contextualized sentence embedding, rather than word-level embeddings. It generates a single fixed-length vector for the entire sentence. An example of the Ada-002 representation for the sentence "*I love this movie.*" can be illustrated in Table 3.8.

Table 3.8 An example of the Ada-002 representation for a sentence

| Sentence | Ada-002 Embedding (Example 1536-D) |
|---|---|
| "I love this movie" | [0.23, -0.45, 0.67, ..., 0.89] |

However, we can use Ada-002 as a word embedding model (instead of sentence embedding). Ada-002 would need to extract embeddings individually for each word in the sentence. The results can be presented as Table 3.9.

Table 3.9 An example of the Ada-002 representation for words

| Words | Ada-002 Embedding (Example 1536-D) |
|---|---|
| I | [0.01, -0.34, 0.67, ..., 0.12] |
| love | [0.56, -0.23, 0.89, ..., -0.02] |
| this | [0.32, 0.11, -0.45, ..., 0.21] |
| movie | [0.78, -0.56, 0.34, ..., 0.09] |

BERT Embedding Representation [100] - BERT's embeddings are derived from a multi-layer transformer architecture, which relies on self-attention mechanisms to model complex linguistic relationships. Each token in a given text sequence is mapped to an embedding that incorporates token-level information, segment embeddings to differentiate sentences, and position embeddings to retain the order of words, which is crucial since transformers do not inherently encode positional information. These embeddings evolve through multiple layers, where lower layers capture syntactic features, middle layers encode general semantic properties, and higher layers refine task-specific knowledge. As a result, BERT embeddings provide a rich representation of language that can be leveraged for various downstream NLP applications. One of the most compelling aspects of BERT embeddings is their adaptability to a wide range of NLP tasks. In text classification, they enhance sentiment analysis, spam detection, and topic modeling by offering a deeper understanding of sentence structures. Named Entity Recognition (NER) benefits from BERT's contextual awareness, improving the accuracy of identifying proper nouns, locations, and organizational entities. BERT embeddings also play a significant role in machine translation, question-answering systems, and semantic similarity tasks, where understanding nuances in meaning is essential. Moreover, search engines and recommendation systems have incorporated BERT to improve text retrieval and

ranking based on contextual relevance. Extracting embeddings from BERT can be approached in different ways, depending on the complexity of the task and available computational resources. Some applications involve fine-tuning the entire BERT model on a domain-specific dataset, which leads to state-of-the-art results but requires substantial computational power. Alternatively, a feature-based approach can be used, where pre-trained BERT embeddings are extracted and fed into a simpler machine learning model, making it a more efficient choice for resource-constrained environments. Typically, embeddings are drawn from the final hidden layer, although some researchers prefer averaging multiple layers to obtain a more generalized representation. The impact of BERT embeddings on NLP has been profound, setting a new standard for contextual word representations and outperforming previous methods in numerous benchmarks. By capturing deep semantic and syntactic relationships within text, BERT enables more accurate and nuanced language understanding. As advancements in transformer models continue, variations like RoBERTa, ALBERT, and DistilBERT have further optimized BERT's performance, efficiency, and scalability. Despite its computational demands, BERT remains a fundamental component of modern NLP research and applications, paving the way for more sophisticated language models in the future.

Suppose an example sentence is "*I love this movie*.", the results can be represented as Figure 3.3.

```
Token    | Embedding (first 5 dimensions shown as an example)
-------------------------------------------------------------
[CLS]    | [0.23, -0.11, 0.45, ..., 0.12]  # (768 values)
I        | [0.51,  0.03, -0.27, ..., 0.68]  # (768 values)
love     | [1.22, -0.76,  0.84, ..., -0.54]  # (768 values)
this     | [-0.12, 0.34, -0.44, ..., 0.09]  # (768 values)
movie    | [0.87, -0.23,  0.61, ..., 1.10]  # (768 values)
[SEP]    | [0.07,  0.14, -0.33, ..., -0.04]  # (768 values)
```

Figure 3.3 Example of BERT embedding representation

The differences among Word2Vec, GloVe, FastText, and Ada-002 can be summarized in Table 3.10. This table highlights key distinctions in their underlying methodologies, training approaches, and applications. While Word2Vec and Gloe generate static word embeddings based on co-occurrence statistics and contextual relationships, FastText enhances this by incorporating subword information, making it more effective for handling rare and out-of-vocabulary words. In contrast, Ada-002, a transformer-based embedding model, provides dynamic and highly contextualized representations, making it well-suited for advanced natural language understanding tasks.

Table 3.10 Summary of the differences among Word2Vec, GloVe, FastText, and Ada-002

| Method | Word Vector Dimension | Embedding Representation | Strengths |
|---|---|---|---|
| Word2Vec | 100-300 | Word-level embeddings (context-based) | It can be used to capture word relationships based on surrounding words. |
| GloVe | 100-300 | Word-level embeddings (co-occurrence-based) | It can be used to capture global statistical relationships between words. |
| FastText | 100-300 | Word + Subword embeddings | It can be used to handle rare/misspelled words effectively. |
| Ada-002 | 1,536 | Sentence-level embedding | It can capture deep contextual meaning of the entire sentence, but it can be used to embed single word as well. |
| BERT embedding | BERT base = 768 | BERT's embedding representation combines token embeddings for individual words or subwords, segment embeddings to distinguish text segments, and position embeddings to retain word order, with the final embedding obtained by summing these components before transformer processing. | BERT embeddings offer contextualized word representations, bidirectional understanding, transfer learning capability, handling of out-of-vocabulary words, sentence-level embeddings, and enhanced semantic comprehension, making them highly effective for various NLP tasks. |

3.3.2.3 Emoticon Representation

In sentiment classification tasks involving multimodal data, emoticons play a crucial role in conveying emotional context that may not be fully captured by text alone. To incorporate emoticons effectively into machine learning models, one-hot encoding is used as a representation method.

One-hot encoding is a simple yet effective technique that converts categorical variables, such as emoticons, into a binary vector format. Each unique emoticon in the dataset is assigned a distinct index, and its corresponding one-hot vector contains a value of 1 at the assigned index and 0 elsewhere. This allows the model to process emoticons as numerical features, making them suitable for sentiment classification.

For example, consider a dataset containing three emoticons: 😊 (happy), 😐 (neutral), and 😢 (sad). Using one-hot encoding, they can be represented as follows:

😊 → [1, 0, 0]

😐 → [0, 1, 0]

😢 → [0, 0, 1]

This representation ensures that each emoticon is treated as an independent feature without implying any ordinal relationship between them. One-hot encoding is particularly useful when dealing with a limited vocabulary of emoticons, as it maintains a clear and interpretable structure. However, in cases where the dataset

contains a large variety of emoticons, alternative embedding techniques such as word embeddings or transformer-based models may be more effective in capturing semantic similarities between different emoticons.

By integrating one-hot encoded emoticon representations with textual features, the sentiment classification model gains a richer and more expressive input representation, leading to improved accuracy in understanding emotional cues in consumer reviews.

3.3.2.4 Combination of Text Representation and Emoticon Representation

This section aims to present an example of combining text representation and emoticon representation, using the consumer review: "*I love this movie* 😊".

To combine text representation and emoticon representation, it needs to represent both components separately and then merge them into a unified representation. Below is how you can achieve this using word embedding for text representation and one-hot encoding for emoticon representation.

*Step 1: Word Representation with Word Embedding* –In this study, three word embedding techniques are used for text representation: Word2Vec, GloVe, FastText, and Ada-002. An example of combining word embedding representation with emoticon representation is presented as follows. However, the process of word embedding is demonstrated step by step based on Word2Vec.

Suppose there is an example consumer review: "*I love this movie* 😊." Each word in the text, "*I love this movie* 😊," is converted into a pre-trained Word2Vec embedding (e.g., 300-dimensional vectors), as illustrated in Table 3.5. To represent the full sentence, the embeddings can be aggregated using methods such as averaging or max-pooling.

$$Text\ representation = \frac{1}{n}\sum_{i=1}^{n} Word\ 2Vec(w_i) \tag{3.1}$$

Example (averaged vector):

[0.26,0.21,0.20,...,0.19]

*Step 2: Emoticon Representation using One-Hot Encoding* - One-hot encoding assigns a binary vector to each emoticon. Suppose we have a predefined set of 3 emoticons. 😊 😐 😵

😊 → [1, 0, 0]

😐 → [0, 1, 0]

😵 → [0, 0, 1]

Since 😊 (smiling face) appears in the review, its one-hot encoded vector is:

[1, 0, 0]

*Step 3: Combining Text and Emoticon Representations* - The final representation concatenates the text and emoticon vectors into a single feature vector:

Table 3.11 Examples of Word2Vec Representation and Emoticon Representation

| Component | Representation |
|---|---|
| Text (Word2Vec Avg.) | [0.26, 0.21, 0.20, ..., 0.19] (300-D) |
| Emoticon (One-Hot Encoding) | [1, 0, 0] (3-D) |

Thus, after concatenating of the word embedding vectors with the one-hot encoded emoticon vectors to form a unified feature representation, the final feature vector becomes:

[0.26,0.21,0.20,...,0.19,1,0,0]

As the examples from Table 3.11, this results in a 303-dimensional feature vector (300 from Word2Vec + 3 from One-Hot Encoding), which can be used as input for machine learning or deep learning models in sentiment classification.

It is important to note that the average vector of word embeddings for GloVe, FastText, and Ada-002 can be calculated using formulas similar to those applied for Word2Vec.

For GloVe word embedding,

$$Text\ representation = \frac{1}{n}\sum_{i=1}^{n} Glove(w_i) \tag{3.2}$$

For FastText word embedding,

$$Text\ representation = \frac{1}{n}\sum_{i=1}^{n} FastText(w_i) \tag{3.3}$$

For Ada-002 applied to word embedding,

$$Text\ representation = \frac{1}{n}\sum_{i=1}^{n} Ada\_002(w_i) \tag{3.4}$$

Key advantages of these approach:

1) Preserves Semantic Meaning – Word embedding captures contextual relationships between words.
2) Retains Emoticon Sentiment – One-hot encoding ensures that emoticons contribute meaningfully to the sentiment analysis.
3) Simple and Efficient – The combination provides a structured yet computationally efficient representation for multimodal sentiment analysis.
4) Feature Fusion: The integration of word embeddings and one-hot encoded emoticons enriches the feature space, enhancing model performance.
5) Word Embedding Variations: Testing multiple embeddings (Word2Vec, GloVe, FastText, and Ada-002) ensures robustness, as each has different

strengths in handling Chinese text.

### 3.3.3 Data Separation using 10-fold Cross Validation

10-fold cross-validation is a technique used in machine learning to evaluate model performance by splitting the dataset into 10 equal parts (folds). The model is trained and tested 10 times, each time using a different fold as the test set while the remaining 9 folds are used for training. The final performance is averaged across all 10 iterations. Steps for 10-Fold Cross-Validation can be detailed as follow.

1) Shuffle the dataset to ensure randomness and prevent order bias.
2) Divide the dataset into 10 equal folds (subsets).
3) Iterate 10 times, where in each iteration:
    - Select one-fold as the validation set.
    - Use the remaining 9-folds as the training set.
    - Train the model and evaluate its performance on the test fold.
4) Compute the final performance score by averaging the results from all 10 iterations.

Benefits of 10-Fold Cross-Validation can be explained as follows.

- More reliable performance estimation: Reduces the risk of biased results by testing on multiple subsets.
- Efficient use of data: Every instance appears in both training and test sets, making it useful for small datasets.
- Reduces overfitting: Provides a balanced evaluation by training on different data splits.
- Less variance in results: Compared to a single train-test split, it ensures stability in performance assessment.

### 3.3.4 Sentiment Classifier Modelling

3.3.4.1 Sentiment Classifier Modelling by Random Forest

Random Forest (RF) is an ensemble learning algorithm that builds multiple decision trees to improve accuracy and reduce overfitting. Steps to develop the RF model can be explained as follows.

1) Initialize the Model: Start with the default settings of Random Forest.
2) Split the Dataset: Divide the data into training and testing sets (e.g., 80:20 split) or use 10-fold cross-validation for robust evaluation.
3) Train the Model: Train Random Forest using the training set.
4) Evaluate Performance: Measure accuracy, F1-score, and AUC on the test set.

To enhance the model's performance, fine-tuning hyperparameters is crucial and this study utilized grid search for fine-tuning hyperparameters. The key hyperparameters include:

- *n_estimators*: The number of decision trees in the Random Forest (commonly set to 100 by default). When using grid search, number of trees in the forest is [100, 200, 300].
- *max_depth*: The maximum depth of each decision tree (setting it too high can cause overfitting). When using grid search, max_depth is [10, 20, None]. "None" means trees grow until all leaves are pure.
- *min_samples_split*: The minimum number of samples required to split a node. When using grid search, min_samples_split is [2, 5, 10].
- *min_samples_leaf*: The minimum number of samples required at a leaf node. When using grid search, min_samples_leaf is [1, 2, 4].
- *max_features*: The number of features considered for the best split (e.g., '*sqrt*' or '*log2*' to prevent overfitting). This work used 'sqrt' because it is suit for classification task.

3.3.4.2 Sentiment Classifier Modelling by SVM

Support Vector Machine (SVM) with a linear kernel is an effective model for text classification, especially when the data is represented in a high-dimensional space, such as word embeddings. Since the text and emoticons are already transformed into numerical vectors, the next step is to train the SVM model. The dataset is divided into 10 subsets (folds). The model is trained 9 times on different combinations of 9-folds and tested on the remaining 1-fold in each iteration. This process is repeated 10 times, and the average performance across all folds is used as the final evaluation metric.

Grid Search is used to optimize C (Regularization Parameter), which controls the trade-off between maximizing the margin and minimizing misclassification errors.

The key hyperparameters to tune in an SVM model with a linear kernel include the C (Regularization Parameter), which controls the trade-off between maximizing the margin and minimizing misclassification errors. A smaller C value allows the model to be more tolerant of misclassifications, resulting in a simpler decision boundary that generalizes well to unseen data, whereas a larger C value forces the model to minimize classification errors, leading to a more complex model that may overfit the training data. Suitable values to test in Grid Search typically include 0.01, 0.1, 1, 10, and 100, allowing the model to balance between bias and variance.

Another important hyperparameter is the loss function, particularly for multiclass classification using a linear SVM. The standard hinge loss function is widely used in SVMs and works by maximizing the margin between classes while penalizing misclassified samples, whereas the squared hinge loss is a variation that imposes a stronger penalty on misclassifications, potentially leading to better

performance in some cases. Both loss functions should be tested during hyperparameter tuning to determine which yields better results for the given dataset.

Additionally, the tolerance (tol) parameter is crucial in controlling the stopping criteria for optimization. Lower values of tol, such as 1e-3, 1e-4, or 1e-5, lead to higher precision but may increase training time, as the model will continue optimizing until convergence is reached.

The maximum iterations (max_iter) parameter is another factor to consider, as it defines the number of iterations the optimization algorithm will perform before stopping. If the model does not converge within the default number of iterations, increasing this value to 1000, 5000, or even 10000 may help ensure proper training without premature termination. Fine-tuning these hyperparameters systematically through Grid Search ensures that the SVM model with a linear kernel is optimized for sentiment classification in multimodal consumer reviews, balancing computational efficiency with classification accuracy.

3.3.4.3 Sentiment Classifier Modelling by CNN

To develop a Convolutional Neural Network (CNN) for sentiment classification, the model architecture must be designed to effectively extract meaningful features from the word embeddings and one-hot encoded emoticons. Since the dataset is multimodal, CNN will primarily focus on capturing spatial relationships within the text embeddings while incorporating emoticon representations. The CNN model for sentiment classification typically consists of the following layers:

Embedding Layer: This layer processes the word embeddings (Word2Vec, GloVe, FastText, or Ada-002). If pre-trained embeddings are used, they can be set as trainable or non-trainable depending on whether fine-tuning is required.

Convolutional Layers: These layers apply multiple filters with different kernel sizes to capture local semantic features within the text.

Activation Function: A ReLU (Rectified Linear Unit) activation is commonly used after convolution to introduce non-linearity.

Max-Pooling Layer: This layer helps to reduce dimensionality while retaining important features.

Flatten Layer: Converts the pooled features into a one-dimensional vector.

Fully Connected (Dense) Layer: This layer integrates extracted features to make sentiment predictions.

Output Layer:

For binary classification (positive/negative), the activation function is sigmoid, outputting a probability between 0 and 1.

For multiclass classification (positive, neutral, negative), the activation function is softmax, producing probability distributions over the three sentiment classes.

The CNN model is trained using 10-fold cross-validation, meaning the dataset is divided into 10 subsets, and training occurs 10 times with different training/testing combinations to ensure robustness.

To optimize the performance of CNN, Grid Search is applied to systematically test different hyperparameter combinations. The key hyperparameters to tune include:

*Number of Filters*: The number of filters in each convolutional layer directly impacts feature extraction. Suitable values include 32, 64, and 128. A higher number of filters can capture more detailed patterns but increases computational cost.

*Kernel Size*: This determines the size of the sliding window in the convolutional operation. Common values to test include 3, 5, and 7, where smaller values focus on short phrases, while larger values capture broader contextual relationships.

*Stride and Padding*: Stride defines how the filter moves across the text, while padding ensures that the output size remains consistent. Typical choices include a stride of 1 or 2 and same padding to maintain input size.

*Pooling Size*: The pooling layer reduces the feature map size, commonly using a pool size of 2 or 3, which helps retain the most significant features while reducing dimensions.

*Dropout Rate*: To prevent overfitting, dropout regularization is applied to randomly deactivate neurons during training. Suitable values include 0.2, 0.3, and 0.5.

*Number of Dense Units*: The fully connected layer determines how learned features are integrated. Suitable values range from 64, 128, to 256 neurons.

*Batch Size*: Defines the number of samples processed before updating model parameters. Common values include 16, 32, and 64, with larger batches requiring more memory but stabilizing training.

*Learning Rate*: The step size for weight updates, typically tested with values like 0.001, 0.0005, and 0.0001. A lower learning rate ensures stable convergence but may slow training.

*Optimizer*: Different optimizers affect convergence speed and accuracy. Common choices include Adam, RMSprop, and SGD (Stochastic Gradient Descent). However, this work used ‘Adam’ as the optimizer.

*3.3.4.4 Sentiment Classifier Modelling by LSTM*

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) that is well-suited for sequential data, such as text. Since your sentiment classification task involves word embeddings (Word2Vec, GloVe, FastText, Ada-002) and one-hot encoded emoticons, LSTM is effective in capturing long-range dependencies and contextual information in the text while integrating emoticon features. The LSTM model for sentiment classification follows a structured architecture:

*Embedding Layer*: This layer processes the word embeddings obtained from Word2Vec, GloVe, FastText, or Ada-002. If pre-trained embeddings are used, they can be set as trainable or frozen based on whether fine-tuning is needed.

*LSTM Layers*: These layers capture sequential dependencies in the text, helping the model understand relationships between words. Typically, one or two stacked LSTM layers are used.

*Dropout Layer*: A dropout mechanism is applied to prevent overfitting by randomly deactivating some neurons during training.

*Fully Connected* (*Dense*) *Layer*: This layer integrates the extracted features for classification.

*Output Layer*:

For binary classification (positive/negative), a sigmoid activation function is used.

For multiclass classification (positive, neutral, negative), a softmax activation function is used to output probabilities across the three sentiment categories.

To improve the model's performance, Grid Search is used to test different configurations of hyperparameters. The key hyperparameters to fine-tune include:

*Number of LSTM Units*: The number of memory units per LSTM cell determines how much past information the model retains. Suitable values include 64, 128, and 256, where larger values enable better learning but increase computational complexity.

*Number of LSTM Layers*: While a single LSTM layer can capture sequential patterns, stacking multiple layers (1, 2, or 3 layers) allows the model to learn deeper representations.

*Dropout Rate*: To reduce overfitting, dropout is applied to randomly deactivate neurons during training. Common values to try include 0.2, 0.3, and 0.5.

*Batch Size*: This determines the number of samples processed before updating the model's weights. Typical values tested are 16, 32, and 64, where smaller batch sizes help capture fine-grained patterns, and larger batch sizes stabilize training.

*Learning Rate*: The step size at which the model updates its parameters, typically tested with values like 0.001, 0.0005, and 0.0001. Lower learning rates improve stability but slow down training.

*Optimizer*: The optimization algorithm affects model convergence and accuracy. Common choices include Adam, RMSprop, and SGD (Stochastic Gradient Descent).

*Recurrent Dropout*: Unlike standard dropout, recurrent dropout is applied within LSTM cells to prevent overfitting in sequence modeling. Values such as 0.1, 0.2, and 0.3 should be tested.

*Hidden Dense Layer Units*: The number of neurons in the fully connected (dense) layer before the output layer. Suitable values are 64, 128, and 256, where larger values help the model integrate more complex features.

*Max Sequence Length*: This defines the maximum number of words considered in each review. Common values tested include 100, 200, and 300 words, based on the average review length.

## 3.4 Improvement and optimization of experimental methods

### 3.4.1 Automatic weight summation

Regarding the model combination method proposed in this paper, this paper attempts to use multiple model feature weighted summation and splicing technology methods to achieve the combination of multiple feature representations.

Table 3.12 Comparison of sentiment analysis using the first dataset for concatenation and automatic weight summation for binary classification

| Model | Representation for Emoticon | 10-fold cross validation evaluation results | | | Evaluation results on the test set | | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | F1-score | AUC | Accuracy | F1-score | AUC |
| Ada-002+GLOVE+RF (Splicing) | One hot encoding | 0.8033 | 0.8015 | 0.8579 | 0.8000 | 0.7954 | 0.8400 |
| Ada-002+GLOVE+RF (Weight sum) | One hot encoding | 0.6968 | 0.6882 | 0.7066 | 0.7500 | 0.7059 | 0.7500 |
| Ada-002+FastText+RF (Splicing) | One hot encoding | 0.8867 | 0.8849 | 0.9359 | 0.9250 | 0.9250 | 0.9925 |
| Ada-002+FastText+RF (Weight sum) | One hot encoding | 0.9058 | 0.9043 | 0.9685 | 0.7767 | 0.6889 | 0.8100 |
| Ada-002+Word2Vec+RF (concatenation) | One hot encoding | 0.8933 | 0.8926 | 0.9395 | 0.9250 | 0.9246 | 0.9775 |
| Ada-002+Word2Vec+RF (weight summation) | One hot encoding | 0.7038 | 0.4585 | 0.7603 | 0.7000 | 0.7805 | 0.8348 |
| Ada-002+BERT+RF (Splicing) | One hot encoding | 0.8729 | 0.8584 | 0.9410 | 0.9250 | 0.9250 | 0.9925 |
| Ada-002+BERT+RF (Weight sum) | One hot encoding | 0.8342 | 0.8154 | 0.9130 | 0.9250 | 0.9250 | 0.9800 |

Note: **RF ( Random Forest ) is used as a classifier, and the other models are used as word embeddings**

Automatic weighting is a method to dynamically adjust the importance of different features in the model. Its core idea is to optimize the feature combination and improve the classification performance of the model by evaluating the contribution of each feature to the model performance and automatically assigning weights.

Step 1: Feature standardization: standardize each feature to ensure that the values of different features are at the same level.

Step 2: Assign initial weights to each feature

Step 3: Normalize the weights to ensure that the sum of the weights is 1.

Step 4: Perform weighted combination of each feature according to the weight.

Step 5: Concatenate the weighted features together to form a comprehensive feature vector.

In the binary sentiment analysis task of the first dataset, the comparison between Splicing and automatic Weight Sum method is shown in Table 3.12. The experimental data show that:

Concatenation method: When Ada-002 is concatenated with FastText, Word2Vec, and BERT, the accuracy, F1-score, and AUC on the 10-fold cross validation and test set are all good, especially the FastText concatenation model with an AUC of 0.9925 on the test set, which is the best performance. This shows that the concatenation method can better capture the features of different embeddings and improve model performance.

Weighted summation method: Although it performs well in cross-validation in some cases (such as Ada-002+FastText+RF), its performance on the test set is significantly reduced. For example, the AUC of the FastText weighted summation model drops from 0.9685 to 0.8100, indicating that this method may have overfitting problems and poor generalization ability.

Overall, the concatenation method performs better in most cases, especially on the test set, where its accuracy, F1-score, and AUC are significantly higher than those of the weighted summation method.

In the binary sentiment analysis task of the second dataset, the comparison between Splicing and automatic Weight Sum method is shown in Table 3.13. The experimental data show that:

The overall performance of the splicing method is more stable: In most cases, the splicing method performs better than the weighted summation on the test set. For example, Ada-002+Word2Vec+RF (spinning) achieves perfect performance on the test set (Accuracy=1.0000, F1-score=1.0000, AUC=1.0000), while the F1-score of the weighted summation method is only 0.9701 and the AUC is 0.8333, indicating that the splicing method has more advantages in capturing features.

Table 3.13 Comparison of sentiment analysis using the second dataset concatenation and automatic weight summation methods for binary classification

| Model | Representation for Emoticon | 10-fold cross validation evaluation results | | | Evaluation results on the test set | | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | F1-score | AUC | Accuracy | F1-score | AUC |
| Ada-002+GLOVE+RF (Splicing) | One hot encoding | 0.9844 | 0.9844 | 0.9990 | 0.9985 | 0.9985 | 0.9999 |
| Ada-002+GLOVE+RF (Weight sum) | One hot encoding | 0.9982 | 0.9981 | 1.0000 | 0.9693 | 0.9701 | 0.9967 |
| Ada-002+ FastText +RF (Splicing) | One hot encoding | 0.9962 | 0.9962 | 0.9997 | 0.9847 | 0.9847 | 0.9985 |
| Ada-002+ FastText +RF (Weight sum) | One hot encoding | 0.9985 | 0.9985 | 1.0000 | 0.9318 | 0.5537 | 0.8148 |
| Ada-002+ Word2Vec +RF (splicing) | One hot encoding | 0.9969 | 0.9969 | 0.9999 | 1.0000 | 1.0000 | 1.0000 |
| Ada-002+ Word2Vec +RF (weight summation) | One hot encoding | 0.9318 | 0.5114 | 0.7731 | 0.9432 | 0.9701 | 0.8333 |
| Ada-002+ BERT +RF (Splicing) | One hot encoding | 0.9985 | 0.9985 | 0.9999 | 0.9877 | 0.9877 | 1.0000 |
| Ada-002+ BERT +RF (Weight sum) | One hot encoding | 0.9946 | 0.9946 | 0.9999 | 1.0000 | 1.0000 | 1.0000 |

Note: **RF ( Random Forest ) is used as a classifier, and the other models are used as word embeddings**

The stability of the weighted summation method is poor: The performance of the weighted summation method varies greatly under different embedding methods. For example, the AUC of Ada-002+FastText+RF (weighted summation) in 10-fold cross validation is 1.0000, but it drops to 0.8148 on the test set, and the F1-score is only 0.5537, indicating that its generalization ability is weak.

The concatenation method performs more stably in the binary sentiment analysis task, especially when combined with FastText, Word2Vec and BERT, the model performance is significantly improved. Although the weighted summation method performs well in some cases, it has poor stability and insufficient generalization ability.

Table 3.14 Comparison of sentiment analysis using the first dataset for concatenation and automatic weight summation for multi -classification

| Model | Representation for Emoticon | 10-fold cross validation evaluation results | | | Evaluation results on the test set | | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | F1-score | AUC | Accuracy | F1-score | AUC |
| Ada-002+ GLOVE +RF (Splicing) | One hot encoding | 0.7902 | 0.7823 | 0.9257 | 0.6136 | 0.5367 | 0.7954 |
| Ada-002+ GLOVE +RF (Weight sum) | One hot encoding | 0.7607 | 0.7501 | 0.9177 | 0.6591 | 0.5748 | 0.8298 |
| Ada-002+ FastText+RF (Splicing) | One hot encoding | 0.8863 | 0.8858 | 0.9444 | 0.9000 | 0.8990 | 0.9612 |
| Ada-002+ FastText+RF (Weight sum) | One hot encoding | 0.9060 | 0.9045 | 0.9542 | 0.7391 | 0.6826 | 0.9417 |
| Ada-002+ Word2Vec+RF (concatenation) | One hot encoding | 0.8933 | 0..8923 | 0.9363 | 0.9000 | 0.8990 | 0.9775 |
| Ada-002+ Word2Vec+RF (weight summation) | One hot encoding | 0.9221 | 0.9171 | 0.9726 | 0.8333 | 0.8074 | 0.9250 |
| Ada-002+ BERT+RF (Splicing) | One hot encoding | 0.9046 | 0.9025 | 0.9567 | 0.9000 | 0.8990 | 0.9975 |
| Ada-002+ BERT+RF (Weight sum) | One hot encoding | 0.9221 | 0.9168 | 0.9863 | 0.7333 | 0.7000 | 0.9196 |

Note: RF ( Random Forest ) is used as a classifier, and the other models are used as word embeddings

In the multi-class sentiment analysis task of the first dataset, the comparison between Splicing and automatic Weight Sum method is shown in Table 3.14. The experimental data show that:

The concatenation method performs more stably on the test set : In most cases, the concatenation method performs better than the weighted summation on the test set. For example, the AUC of Ada-002+FastText+RF (concatenation) on the test set is 0.9612, while the weighted summation method is only 0.9417. In addition, the AUC of Ada-002+BERT+RF (concatenation) reaches 0.9975, which is significantly higher than the weighted summation of 0.9196, indicating that the concatenation method has more advantages in multi-classification tasks.

The weighted sum method performs better in 10-fold cross validation, but is unstable on the test set : The weighted sum method performs better than the concatenation method in 10-fold cross validation. For example, the AUC of Ada-002+Word2Vec+RF (weighted sum) in 10-fold cross validation is 0.9726, which is higher than the 0.9363 of the concatenation method. However, its AUC on the test set drops to 0.9250, indicating that its generalization ability is weak.

In general, the concatenation method performs more stably in the three-class sentiment analysis task, especially when combined with FastText and BERT, the model performance is significantly improved. Although the weighted summation method performs well in the 10-fold cross validation, it has poor stability on the test set and insufficient generalization ability.

Table 3.15 Comparison of sentiment analysis using the second dataset for concatenation and automatic weight summation for multi -classification

| Model | Representation for Emoticon | 10-fold cross validation evaluation results | | | Evaluation results on the test set | | |
|---|---|---|---|---|---|---|---|
| | | Accuracy | F1-score | AUC | Accuracy | F1-score | AUC |
| Ada-002+GLOVE+RF (Splicing) | One hot encoding | 0.9831 | 0.9833 | 0.9988 | 0.8100 | 0.3671 | 0.7681 |
| Ada-002+GLOVE+RF (Weight sum) | One hot encoding | 0.9826 | 0.9827 | 0.9987 | 0.8150 | 0.3888 | 0.7546 |
| Ada-002+FastText+RF (Splicing) | One hot encoding | 0.9923 | 0.9923 | 0.9997 | 0.9969 | 0.9969 | 1.0000 |
| Ada-002+FastText+RF (Weight sum) | One hot encoding | 0.9965 | 0.9965 | 1.0000 | 0.9318 | 0.9051 | 0.8730 |
| Ada-002+Word2Vec+RF (concatenation) | One hot encoding | 0.9954 | 0.9954 | 0.9999 | 1.0000 | 1.0000 | 1.0000 |
| Ada-002+Word2Vec+RF (weight summation) | One hot encoding | 0.9945 | 0.9944 | 0.9999 | 0.9943 | 0.9785 | 1.0000 |
| **Ada-002+BERT+RF (Splicing)** | One hot encoding | **0.9939** | **0.9938** | **0.9995** | **0.9969** | **0.9969** | **1.0000** |
| **Ada-002+BERT+RF (Weight sum)** | One hot encoding | **0.9985** | **0.9985** | **1.0000** | **0.9432** | **0.7073** | **0.9281** |

Note: RF ( Random Forest ) is used as a classifier, and the other models are used as word embeddings

In the multi-class sentiment analysis task of the secondt dataset, the comparison between Splicing and automatic Weight Sum method is shown in Table 3.15. The experimental data show that:

The splicing method performs better on the test set : In most cases, the splicing method performs better than the weight summation on the test set. For example, the AUC of Ada-002+FastText+RF (splicing) and Ada-002+Word2Vec+RF (splicing) on the test set both reached 1.0000, showing perfect classification ability. Among the weight summation methods, the AUC of Ada-002+FastText+RF (weight

summation) is only 0.8730, indicating that the splicing method has more advantages in multi-classification tasks.

The weighted sum method performs well in 10-fold cross validation, but is unstable on the test set . The weighted sum method performs close to the concatenation method in 10-fold cross validation. For example, Ada-002+BERT+RF (weighted sum) has an AUC of 1.0000 in 10-fold cross validation, which is comparable to the concatenation method. However, its AUC on the test set drops to 0.9281, and its F1 score is only 0.7073, indicating that its generalization ability is weak.

The concatenation method performs better in multi-classification sentiment analysis tasks, especially when combined with FastText and Word2Vec, the model performance is significantly improved.

From the experimental data in Table 3.12-3.15, we can see that although the concatenation method has a higher dimension, it retains richer original information, allowing the downstream classifier to learn the best feature combination method by itself, so it usually performs better. Although the automatic weight summation reduces the dimension, it may discard valuable information too early, resulting in a worse effect than the concatenation method. Therefore, this paper finally chooses to use the concatenation method for feature fusion.

From the implementation data, it can be seen that the overall performance of the three-model combination embedding method is excellent, which is significantly higher than the two-model combination method, especially after the three-group feature combination of Ada-002+BERT+emoji, and then through the RF (Random Forest) classifier, the overall output performance index is the best.

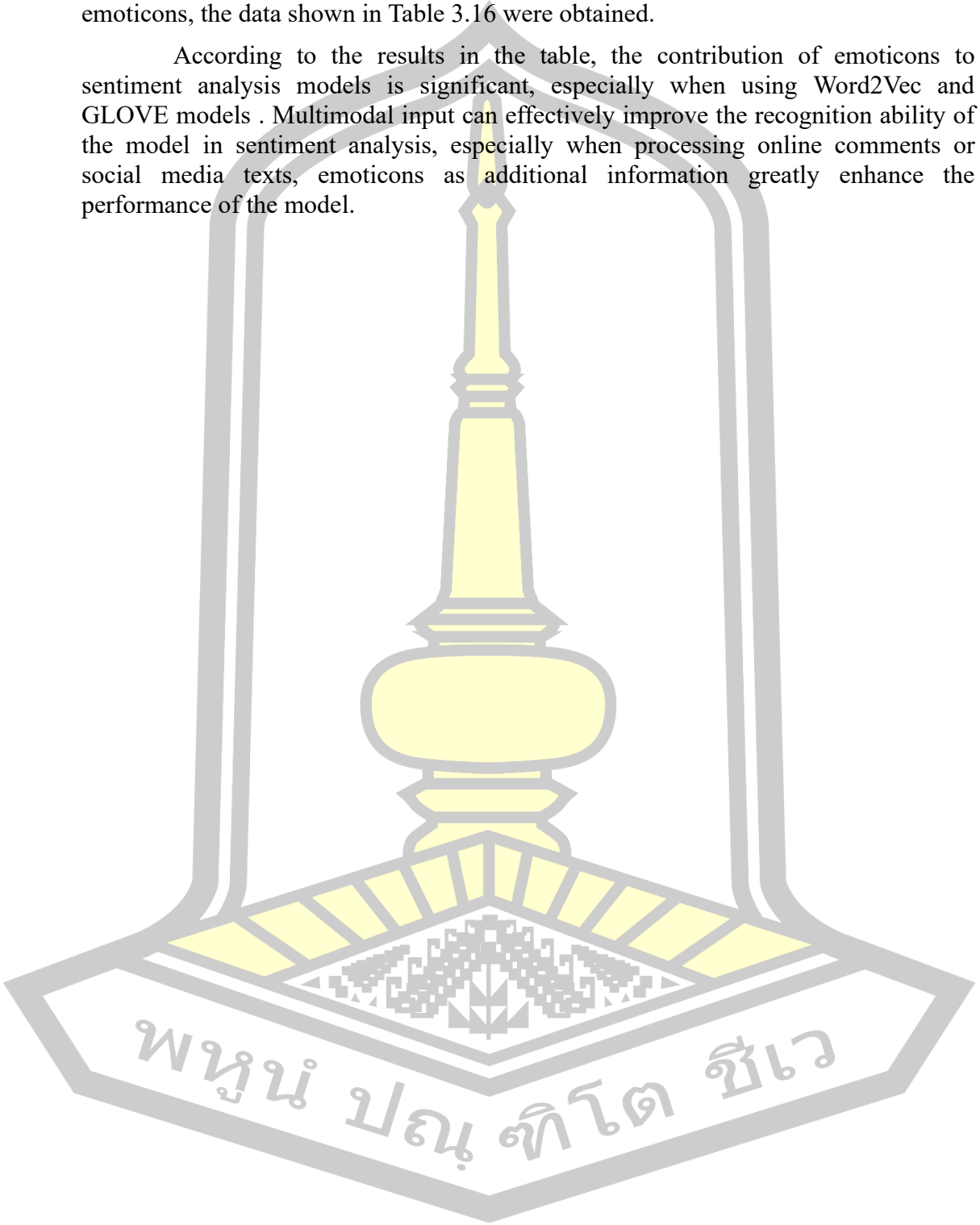
### 3.4.2 Comparison of sentiment analysis using the first dataset text and text + emoticons

Table 3.16 Comparative analysis of single-modal and multi-modal prediction effects

| Model | Precision | Recall | F1 Score | Accuray | MCC | AUC |
|---|---|---|---|---|---|---|
| Word2Vec + CNN (text) | 0.83 | 0.80 | 0.80 | 0.80 | 0.63 | 0.89 |
| Word2Vec + CNN ( text + emoji) | 0.83 | 0.86 | 0.84 | 0.83 | 0.65 | 0.89 |
| GLOVE+CNN (text) | 0.81 | 0.80 | 0.80 | 0.80 | 0.61 | 0.84 |
| GLOVE+CNN (text+emoji) | 0.82 | 0.86 | 0.84 | 0.83 | 0.66 | 0.89 |
| FastText + CNN (text) | 0.70 | 0.70 | 0.70 | 0.70 | 0.40 | 0.76 |
| FastText + CNN (text + emoji) | 0.79 | 0.79 | 0.78 | 0.79 | 0.58 | 0.86 |
| Word2Vec + LSTM ( Text) | 0.84 | 0.74 | 0.78 | 0.80 | 0.60 | 0.91 |
| Word2Vec + LSTM ( text + emoji) | 0.86 | 0.79 | 0.82 | 0.82 | 0.64 | 0.91 |
| GLOVE+ LSTM (Text) | 0.26 | 0.48 | 0.32 | 0.47 | 0.11 | 0.87 |
| GLOVE+ LSTM (text+emoji) | 0.82 | 0.71 | 0.76 | 0.79 | 0.55 | 0.90 |
| FastText + LSTM (Text) | 0.73 | 0.74 | 0.73 | 0.74 | 0.47 | 0.85 |
| FastText + LSTM (text + emoji) | 0.83 | 0.69 | 0.75 | 0.77 | 0.55 | 0.88 |

In the binary classification, single-modal and multi-modal sentiment prediction experimental tests were conducted. For two input types, text and text + emoticons, the data shown in Table 3.16 were obtained.

According to the results in the table, the contribution of emoticons to sentiment analysis models is significant, especially when using Word2Vec and GLOVE models . Multimodal input can effectively improve the recognition ability of the model in sentiment analysis, especially when processing online comments or social media texts, emoticons as additional information greatly enhance the performance of the model.



# CHAPTER 4 EXPERIMENTAL RESULTS AND DISCUSSION

This chapter presents the experimental results and a critical analysis of the findings. It includes a detailed evaluation of the models used in the study, comparing their performance in sentiment classification. The discussion highlights the strengths and limitations of each model, providing insights into their effectiveness in handling multimodal consumer reviews.

This study investigates sentiment classification in multimodal consumer reviews by integrating textual representations with emoticons through data fusion. The results demonstrate that combining word embeddings with one-hot encoding for emoticons enhances sentiment classification performance across multiple machine learning models.

Five text representation techniques—Word2Vec, GloVe, FastText, Ada-002, and BERT embedding—were applied to capture the semantic features of textual reviews. The presence of emoticons within the dataset introduces an additional layer of sentiment representation. By applying one-hot encoding, emoticons were treated as discrete categorical features that complement textual data. This approach enhances the model's ability to capture sentiment cues that may not be explicitly conveyed through text alone.

## 4.1 Evaluation Results for Binary-based Sentiment Classification for Multimodal Consumer Reviews

To assess the performance of the binary sentiment classification models developed using the proposed method, 10-fold cross-validation was performed to ensure a rigorous and reliable evaluation. The results, summarized in Table 4.1 and Table 4.2, provide a comprehensive comparison of model performance across multiple folds. These tables present key evaluation metrics, including accuracy, F1-score, and AUC, offering valuable insights into the effectiveness and robustness of each model in analyzing multimodal consumer reviews.

### 4.1.1 Evaluation Results from 10-Fold Cross-Validation for Binary-based Sentiment Classification using the First Dataset

This section presents the experimental results of the evaluation conducted using 10-fold cross-validation for binary sentiment classification on the first dataset. The results can be presented as Table 4.1.

This section evaluates the effectiveness of various machine learning and deep learning models in binary sentiment classification of multimodal movie reviews, which consist of textual content and emoticons. The results illustrate the impact of different text representation techniques, namely Word2Vec, GloVe, FastText, Ada-002, and BERT embedding, in combination with one-hot encoding for emoticons. The analysis reveals clear performance distinctions across models, emphasizing the

significance of choosing appropriate text representation techniques and classification algorithms for sentiment analysis.

Table 4.1 Evaluation Results from 10-Fold Cross-Validation for Binary-based Sentiment Classification using the First Dataset

| Models | Representation for Text | Representation for Emoticon | Accuracy | F1-score | AUC |
|---|---|---|---|---|---|
| Random Forest | Word2Vec | One hot encoding | 0.7182 | 0.6923 | 0.7935 |
| | GloVE | One hot encoding | 0.7491 | 0.7197 | 0.8130 |
| | FastText | One hot encoding | 0.7473 | 0.7262 | 0.8584 |
| | Ada-002 | One hot encoding | 0.7791 | 0.7648 | 0.9455 |
| | BERT Embedding | One hot encoding | 0.8000 | 0.7995 | 0.8965 |
| SVM with Linear | Word2Vec | One hot encoding | 0.7955 | 0.7665 | 0.8850 |
| | GloVE | One hot encoding | 0.8053 | 0.7983 | 0.9108 |
| | FastText | One hot encoding | 0.7773 | 0.7680 | 0.8343 |
| | Ada-002 | One hot encoding | 0.9242 | 0.9214 | 0.9790 |
| | BERT Embedding | One hot encoding | 0.7788 | 0.7704 | 0.8085 |
| CNN | Word2Vec | One hot encoding | 0.7970 | 0.7844 | 0.8988 |
| | GloVE | One hot encoding | 0.8492 | 0.8404 | 0.8975 |
| | FastText | One hot encoding | 0.8386 | 0.8325 | 0.8893 |
| | Ada-002 | One hot encoding | 0.8645 | 0.8396 | 0.9670 |
| | BERT Embedding | One hot encoding | 0.7538 | 0.7453 | 0.8270 |
| LSTM | Word2Vec | One hot encoding | 0.7447 | 0.6995 | 0.9067 |
| | GloVE | One hot encoding | 0.7523 | 0.7115 | 0.8860 |
| | FastText | One hot encoding | 0.7253 | 0.6911 | 0.8580 |
| | Ada-002 | One hot encoding | 0.8835 | 0.8801 | 0.9491 |
| | BERT Embedding | One hot encoding | 0.7681 | 0.7509 | 0.8750 |

Among the machine learning models, the Support Vector Machine (SVM) with a linear kernel demonstrated the highest performance when utilizing Ada-002 for text representation, achieving an accuracy of 0.9242, an F1-score of 0.9214, and an AUC of 0.9790. The superior performance of this model can be attributed to the ability of Ada-002 to generate semantically rich embeddings that align well with the linear separability assumption of SVM. The model's robustness in handling high-dimensional feature spaces, along with the enhanced representation from Ada-002, significantly contributed to its high classification performance. On the other hand, Random Forest (RF) showed relatively lower performance compared to SVM, even when using Ada-002, with an accuracy of 0.7791, an F1-score of 0.7648, and an AUC of 0.9455. This outcome aligns with expectations since Random Forest relies on ensemble decision trees, which may struggle to fully leverage deep contextual representations provided by Ada-002. Although RF performed adequately, its decision boundaries appear less effective in capturing the nuanced sentiment present in the multimodal dataset.

For deep learning models, the Convolutional Neural Network (CNN) with Ada-002 emerged as the best-performing model, achieving an accuracy of 0.8645, an F1-score of 0.8396, and an AUC of 0.9670. The CNN's ability to capture local dependencies and hierarchical structures in textual data played a crucial role in its success, while Ada-002 further refined feature representations. The one-hot encoding for emoticons complemented CNN's capacity for pattern recognition, reinforcing sentiment cues embedded within the reviews. In contrast, the Long Short-Term Memory (LSTM) network yielded the lowest performance among deep learning

models, particularly when using FastText for text representation, with an accuracy of 0.7253, an F1-score of 0.6911, and an AUC of 0.8580. This suboptimal performance likely stems from the sequential nature of LSTM, which may not be as effective when dealing with short, highly variable review texts. Moreover, FastText, while efficient in capturing subword-level information, may introduce excessive noise in sentiment classification tasks, leading to a reduction in classification performance.

The variation in performance across text representation techniques further underscores the importance of selecting an appropriate embedding approach. Ada-002 consistently outperformed conventional embeddings such as Word2Vec, GloVe, FastText, and even BERT embedding across all models. This suggests that the deep contextualized representations generated by Ada-002 provide superior feature embeddings that enhance sentiment classification accuracy. Notably, BERT embedding did not perform as well as Ada-002, particularly in the SVM and CNN models. While BERT is highly effective in capturing deep contextual relationships, its application to short movie reviews with high structural and sentiment variability may not be optimal. Additionally, concatenation-based fusion with one-hot encoded emoticons may not have fully complemented BERT's representations, thereby limiting its effectiveness in this specific experimental setup.

The incorporation of one-hot encoding for emoticons positively influenced classification performance across all models, providing an additional layer of sentiment information that enriched feature representations. Emoticons serve as explicit sentiment markers, reinforcing the textual content and improving model predictions. This effect was particularly noticeable in CNN and SVM, where structured feature extraction and hyperplane-based classification effectively leveraged the multimodal data fusion.

Overall, the findings from this section indicate that SVM with Ada-002 yields the best performance among machine learning models, while CNN with Ada-002 achieves the highest accuracy among deep learning models. In contrast, LSTM with FastText performs the worst, likely due to the sequential nature of LSTM and the subword-based characteristics of FastText. Additionally, the study demonstrates that the inclusion of one-hot encoding for emoticons contributes positively to sentiment classification, particularly when combined with robust text representations. These insights highlight the importance of selecting appropriate text representations and model architectures for sentiment classification in multimodal contexts. Future research may explore alternative fusion strategies beyond concatenation to further enhance performance and better integrate multimodal sentiment information.

### 4.1.2 Evaluation Results from 10-Fold Cross-Validation for Binary-based Sentiment Classification using the Second Dataset

This section presents the experimental results of the evaluation conducted using 10-fold cross-validation for binary sentiment classification on the second dataset. The results can be presented as Table 4.2.

Table 4.2 Evaluation Results from 10-Fold Cross-Validation for Binary-based Sentiment Classification using the Second Dataset

| Models | Representation for Text | Representation for Emoticon | Accuracy | F1-score | AUC |
|---|---|---|---|---|---|
| Random Forest | Word2Vec | One hot encoding | 0.9603 | 0.9602 | 0.9925 |
| | GloVE | One hot encoding | 0.9611 | 0.9610 | 0.9948 |
| | FastText | One hot encoding | 0.9574 | 0.9753 | 0.9972 |
| | Ada-002 | One hot encoding | 0.9786 | 0.9783 | 0.9976 |
| | BERT Embedding | One hot encoding | 0.9956 | 0.9956 | 0.9999 |
| SVM with Linear | Word2Vec | One hot encoding | 0.9147 | 0.9135 | 0.8810 |
| | GloVE | One hot encoding | 0.5596 | 0.4340 | 0.5555 |
| | FastText | One hot encoding | 0.9094 | 0.9078 | 0.8521 |
| | Ada-002 | One hot encoding | 0.9830 | 0.9829 | 0.9972 |
| | BERT Embedding | One hot encoding | 0.9583 | 0.9580 | 0.9691 |
| CNN | Word2Vec | One hot encoding | 0.9903 | 0.9903 | 0.9994 |
| | GloVE | One hot encoding | 0.9879 | 0.9878 | 0.9975 |
| | FastText | One hot encoding | 0.9860 | 0.9859 | 0.9974 |
| | Ada-002 | One hot encoding | 0.9897 | 0.9893 | 0.9999 |
| | BERT Embedding | One hot encoding | 0.9878 | 0.9877 | 0.9993 |
| LSTM | Word2Vec | One hot encoding | 0.9593 | 0.9559 | 0.9978 |
| | GloVE | One hot encoding | 0.9603 | 0.9569 | 0.9979 |
| | FastText | One hot encoding | 0.9568 | 0.9550 | 0.9910 |
| | Ada-002 | One hot encoding | 0.9786 | 0.9780 | 0.9999 |
| | BERT Embedding | One hot encoding | 0.9834 | 0.9830 | 0.9999 |

In Table 4.2, among the machine learning models evaluated, Random Forest consistently achieved high classification performance, especially when using BERT embeddings for text representation. The model attained an accuracy of 0.9956, an F1-score of 0.9956, and an AUC of 0.9999, demonstrating its ability to leverage BERT's powerful contextual embeddings to distinguish sentiment effectively. Random Forest, as an ensemble-based model, benefits from combining multiple decision trees, which enhances its robustness and mitigates overfitting. The effectiveness of BERT embeddings, which incorporate bidirectional contextual representations, further strengthens the model's ability to capture nuanced sentiment patterns within the dataset. Ada-002 also performed well when paired with Random Forest, achieving an accuracy of 0.9786 and an AUC of 0.9976, reinforcing the advantage of deep contextualized embeddings in sentiment classification.

Conversely, the lowest-performing machine learning model was SVM with GloVe, which resulted in an accuracy of 0.5596, an F1-score of 0.4340, and an AUC of 0.5555. This significant drop in performance suggests that GloVe's static word embeddings were inadequate in capturing the sentiment variations present in the dataset, while SVM's reliance on linear decision boundaries limited its ability to effectively classify complex sentiment expressions. Unlike tree-based models such as Random Forest, which can effectively handle non-linearly separable data, SVM struggled to distinguish sentiment classes when using GloVe's fixed-word vector representations, leading to suboptimal classification results.

In the context of deep learning models, CNN emerged as the top-performing model, particularly when using Word2Vec or Ada-002 for text representation. The CNN model achieved an accuracy of 0.9903, an F1-score of 0.9903, and an AUC of 0.9994 when combined with Word2Vec, while comparable results were observed for CNN with Ada-002, BERT, and other advanced embeddings. CNN's ability to capture hierarchical patterns and local dependencies in text contributed significantly to its superior performance, allowing it to detect sentiment-related structures within short-form reviews. The inclusion of one-hot encoding for emoticons further reinforced CNN's predictive capability by providing an additional layer of sentiment cues. The effectiveness of CNN can be attributed to its convolutional filters, which excel at identifying key features in text data, making it particularly suitable for sentiment classification.

LSTM also demonstrated strong performance, albeit slightly lower than CNN. The best LSTM model, which utilized BERT embeddings, achieved an accuracy of 0.9834, an F1-score of 0.9830, and an AUC of 0.9999, showing that sequential models remain highly effective when trained with rich text representations. However, compared to CNN, LSTM may not fully capitalize on sentiment cues embedded in short reviews due to its sequential dependency modeling. While LSTM excels in handling long-term dependencies, it may introduce unnecessary complexity when processing shorter review texts. This limitation likely accounts for its slightly lower performance relative to CNN.

The comparison of text representation techniques highlights the significance of selecting an appropriate embedding approach. Ada-002 and BERT consistently outperformed traditional embeddings such as Word2Vec, GloVe, and FastText across all models. Ada-002's strong performance suggests that its deep learning-based embeddings effectively capture contextual nuances, while BERT's bidirectional nature allows for richer sentiment representation. On the other hand, GloVe's inconsistent performance, particularly with SVM, highlights its limitation in capturing sentiment polarity in highly contextualized review texts. The relatively static nature of GloVe embeddings may hinder their ability to differentiate sentiment nuances, which is a crucial factor in sentiment classification.

The incorporation of one-hot encoding for emoticons played a pivotal role in enhancing classification performance across all models by enriching the feature space with additional sentiment indicators. Emoticons function as explicit sentiment markers that complement textual content, providing a more comprehensive sentiment representation. This effect was especially evident in CNN and Random Forest, where structured feature extraction mechanisms and ensemble-based learning strategies benefited significantly from the inclusion of multimodal data. The results affirm the importance of integrating textual and non-textual features in sentiment analysis, demonstrating that multimodal fusion can improve classification accuracy.

Overall, the findings of this study suggest that CNN with Word2Vec or Ada-002 delivers the best performance among deep learning models, while Random Forest with BERT embedding achieves the highest accuracy among machine learning models. The lowest-performing model was SVM with GloVe, which struggled due to the linear nature of SVM and the limitations of static word embeddings. These results emphasize the necessity of selecting suitable text representations and classification architectures to optimize sentiment classification performance in multimodal contexts.

Future research may explore alternative fusion techniques beyond concatenation, such as attention-based methods or transformer-based fusion mechanisms, to further enhance the integration of textual and emoticon-based sentiment cues.

### 4.1.3 Summary for Binary-based Sentiment Classification on Multimodal Consumer Reviews

The evaluation results from 10-fold cross-validation for binary sentiment classification using both datasets highlight the performance variations among machine learning and deep learning models, emphasizing the role of different text representation techniques and the impact of one-hot encoding for emoticons in multimodal sentiment analysis. The first dataset demonstrated that SVM with a linear kernel, when combined with Ada-002 embeddings, yielded the highest classification performance among machine learning models, achieving an accuracy of 0.9242, an F1-score of 0.9214, and an AUC of 0.9790. The superior performance of this combination can be attributed to Ada-002's ability to generate semantically rich and contextually relevant embeddings, which align well with SVM's requirement for a well-defined margin in high-dimensional feature spaces. In contrast, Random Forest, while still performing adequately with an accuracy of 0.7791, showed lower effectiveness in capturing deep contextual representations, as its decision-tree-based approach is less adept at handling high-dimensional embeddings.

For deep learning models on the first dataset, CNN with Ada-002 emerged as the best-performing model, achieving an accuracy of 0.8645, an F1-score of 0.8396, and an AUC of 0.9670. CNN's ability to extract hierarchical features from text, coupled with Ada-002's superior word representations, enabled effective sentiment classification. The inclusion of one-hot encoding for emoticons further enhanced CNN's capability by providing additional sentiment cues that complemented the text-based feature extraction process. In contrast, LSTM with FastText recorded the lowest performance, with an accuracy of 0.7253, an F1-score of 0.6911, and an AUC of 0.8580. The sequential nature of LSTM, which depends on long-range dependencies, likely struggled with shorter, highly variable review texts. FastText, while proficient in capturing subword-level information, introduced excessive noise, leading to reduced classification accuracy.

In the second dataset, Random Forest with BERT embeddings outperformed all other machine learning models, achieving an accuracy of 0.9956, an F1-score of 0.9956, and an AUC of 0.9999. The effectiveness of BERT stems from its deep bidirectional context modeling, which provides nuanced sentiment representations that Random Forest effectively leveraged through its ensemble learning approach. The worst-performing model among machine learning algorithms was SVM with GloVe, which recorded an accuracy of 0.5596, an F1-score of 0.4340, and an AUC of 0.5555. GloVe's static nature failed to capture contextual sentiment variations effectively, and SVM's reliance on linear decision boundaries was insufficient for handling complex, non-linearly separable sentiment expressions.

Among deep learning models on the second dataset, CNN with Word2Vec produced the highest accuracy, reaching 0.9903, with an F1-score of 0.9903 and an AUC of 0.9994. CNN's ability to capture local dependencies and hierarchical patterns in text, combined with the strong word associations embedded in Word2Vec

representations, facilitated robust sentiment classification. The integration of emoticons through one-hot encoding further enhanced CNN's performance by supplementing textual sentiment with additional emotional markers. LSTM with BERT also performed well, with an accuracy of 0.9834, an F1-score of 0.9830, and an AUC of 0.9999. However, compared to CNN, LSTM's sequential dependency modeling introduced computational overhead without a proportional gain in performance, making CNN a more efficient choice for this sentiment classification task.

Overall, the results suggest that the highest-performing models across both datasets were Random Forest with BERT among machine learning models and CNN with Ada-002 or Word2Vec among deep learning models. These models effectively utilized their respective embedding techniques to capture sentiment nuances and classify reviews with high precision. The lowest-performing models were SVM with GloVe and LSTM with FastText, primarily due to the limitations of their respective text representations and classification mechanisms. The findings underscore the importance of selecting the right combination of embedding techniques and model architectures for sentiment analysis in multimodal contexts, while also highlighting the effectiveness of multimodal fusion techniques, such as one-hot encoding for emoticons, in enhancing classification accuracy. Future research could further explore alternative fusion techniques beyond concatenation, such as attention-based mechanisms or transformer-based fusion, to refine the integration of textual and emoticon-based sentiment cues.

## 4.2 Evaluation Results for Multiclass-based Sentiment Classification for Multimodal Consumer Reviews

To evaluate the performance of the multiclass sentiment classification models developed using the proposed method, 10-fold cross-validation was conducted to ensure a thorough and reliable assessment. The results, presented in Table 4.3 and Table 4.4, offer a detailed comparison of model performance across multiple folds. These tables highlight key metrics such as accuracy, F1-score, and AUC, providing valuable insights into the effectiveness and stability of each model in classifying multimodal consumer reviews.

### 4.2.1 Evaluation Results from 10-Fold Cross-Validation for Multiclass-based Sentiment Classification using the First Dataset

This section presents the experimental results of the evaluation conducted using 10-fold cross-validation for multiclass sentiment classification on the first dataset. The results can be presented as Table 4.3.

Table 4.3 Evaluation Results from 10-Fold Cross-Validation for Multiclass-based Sentiment Classification using the First Dataset

| Models | Representation for Text | Representation for Emoticon | Accuracy | F1-score | AUC |
|---|---|---|---|---|---|
| Random Forest | Word2Vec | One hot encoding | 0.5859 | 0.5708 | 0.8260 |
| | GloVE | One hot encoding | 0.5859 | 0.5713 | 0.8145 |
| | FastText | One hot encoding | 0.6036 | 0.5909 | 0.7796 |
| | Ada-002 | One hot encoding | 0.7833 | 0.7852 | 0.9151 |
| | BERT Embedding | One hot encoding | 0.5641 | 0.5459 | 0.7852 |
| SVM with Linear | Word2Vec | One hot encoding | 0.6171 | 0.5772 | 1.0000 |
| | GloVE | One hot encoding | 0.6274 | 0.5899 | 1.0000 |
| | FastText | One hot encoding | 0.6012 | 0.5581 | 0.9467 |
| | Ada-002 | One hot encoding | 0.7500 | 0.7411 | 0.9217 |
| | BERT Embedding | One hot encoding | 0.7788 | 0.7704 | 0.8085 |
| CNN | Word2Vec | One hot encoding | 0.5676 | 0.5451 | 0.7535 |
| | GloVE | One hot encoding | 0.5549 | 0.4954 | 0.7624 |
| | FastText | One hot encoding | 0.4997 | 0.4790 | 0.7001 |
| | Ada-002 | One hot encoding | 0.5432 | 0.4697 | 0.7086 |
| | BERT Embedding | One hot encoding | 0.6405 | 0.6332 | 0.8729 |
| LSTM | Word2Vec | One hot encoding | 0.6042 | 0.5758 | 0.8243 |
| | GloVE | One hot encoding | 0.6288 | 0.5971 | 0.8582 |
| | FastText | One hot encoding | 0.6185 | 0.5851 | 0.8278 |
| | Ada-002 | One hot encoding | 0.6060 | 0.5930 | 0.7994 |
| | BERT Embedding | One hot encoding | 0.7875 | 0.7787 | 0.9437 |

In Table 4.3, the results indicate that contextual word embeddings, particularly Ada-002 and BERT, generally lead to improved sentiment classification performance compared to traditional word embeddings like Word2Vec, GloVe, and FastText. This finding aligns with the theoretical underpinnings of pre-trained transformer-based embeddings, which capture rich contextual meanings, allowing models to interpret words in relation to their surrounding text.

Among the traditional embeddings, GloVe performed slightly better than Word2Vec and FastText, likely due to its ability to capture global word co-occurrence information. However, the static nature of these embeddings, where words have fixed representations regardless of context, limits their ability to adapt to different sentiment expressions. This contrasts with BERT and Ada-002, which leverage deep transformer architectures to provide context-dependent embeddings, significantly improving sentiment classification performance.

One notable aspect of this study is the inclusion of one-hot encoding for emoticons, which introduces a multimodal element to the dataset. While emoticons serve as effective sentiment indicators in text-based communication, their impact on classification performance depends on the model's ability to integrate them with textual features. The concatenation-based data fusion method employed in this study combines text embeddings with one-hot encoded emoticons, enabling models to leverage both modalities. However, since one-hot encoding treats each emoticon as an independent feature without considering semantic relationships among them, its effectiveness remains limited compared to more sophisticated multimodal fusion techniques such as attention-based fusion or vector-space embeddings for emoticons.

The evaluation of Random Forest, SVM with a linear kernel, CNN, and LSTM reveals significant variations in classification effectiveness, demonstrating the inherent strengths and weaknesses of each model.

SVM with a linear kernel consistently outperformed other models when paired with BERT and Ada-002 embeddings, achieving an accuracy of 0.7780 and an F1-score of 0.7704 with BERT embeddings. This superior performance is well-supported by the theoretical properties of SVM in high-dimensional spaces, where it finds the optimal hyperplane to separate sentiment classes effectively. The ability of linear SVM to work well with pre-trained contextual embeddings suggests that sentiment classification benefits from strong feature representations, making feature engineering more critical than model complexity in certain cases.

The LSTM model with BERT embeddings emerged as the best-performing model, achieving an accuracy of 78.75% and an F1-score of 0.7787, along with the highest AUC of 0.9437. This result is consistent with LSTM's ability to capture sequential dependencies in textual data, allowing it to leverage context-aware embeddings more effectively than other models. Given that sentiment classification often involves long-range dependencies, the combination of BERT embeddings with LSTM's recurrent architecture enables the model to discern subtle shifts in sentiment more accurately.

Random Forest, a tree-based ensemble method, exhibited mixed performance across different embeddings. While it performed well with Ada-002 embeddings (accuracy at 0.7833, F1-score at 0.7852, and AUC at 0.9151), its performance dropped significantly when paired with BERT embeddings (accuracy at 0.5641 and F1-score at 0.5459). This discrepancy suggests that tree-based models struggle to fully exploit the advantages of contextual embeddings, likely due to their reliance on discrete decision boundaries rather than continuous latent representations. Additionally, Random Forest models tend to work better with structured numerical data than high-dimensional text embeddings, explaining their variability in performance.

CNN demonstrated the lowest overall performance, particularly when combined with FastText embeddings, where it achieved an accuracy of only 0.4997 and an F1-score of 0.4790. The primary reason for CNN's poor performance lies in its reliance on convolutional filters to extract local features (n-grams), which are less effective for understanding long-term dependencies in sentiment classification. Unlike image processing tasks where CNNs excel, sentiment analysis requires models that can capture the contextual relationships between words across a sentence, something that CNN's fixed window size fails to achieve effectively. Moreover, CNN's inability to model sequential dependencies means that even when using contextual embeddings, its performance does not improve significantly.

A direct comparison of the LSTM model with BERT embeddings (best-performing model) and CNN with FastText embeddings (worst-performing model) reveals key insights into the characteristics that contribute to effective sentiment classification.

*Sequential Learning vs. Local Feature Extraction*: LSTM's recurrent architecture allows it to retain and process information across long sequences, making

it particularly effective for sentiment classification, where context matters. CNN, by contrast, applies convolutional filters to fixed-length text windows, making it unsuitable for capturing long-range dependencies in sentiment expressions.

*Contextual Embeddings vs. Static Embeddings*: BERT embeddings provide dynamic word representations, meaning that the same word can have different vector representations depending on the context in which it appears. This significantly enhances sentiment classification by ensuring that nuanced differences in meaning are captured. FastText, despite handling subword information, remains context-agnostic, meaning that it fails to differentiate between words used in positive or negative contexts, leading to lower performance in sentiment classification.

*Impact of Emoticon Encoding*: While one-hot encoding of emoticons provides a basic multimodal representation, its impact was more pronounced in models that effectively integrate multimodal features (such as LSTM). CNN, with its spatial feature extraction approach, was less effective at leveraging discrete categorical features like one-hot encoded emoticons, further contributing to its poor classification performance.

In summary of evaluation results for multiclass-based sentiment classification for multimodal consumer reviews, the findings from this study reinforce the importance of choosing the right text representation and classifier for sentiment analysis. Transformer-based embeddings (BERT and Ada-002) consistently outperformed traditional embeddings, demonstrating their ability to capture rich contextual information. Among the classifiers, SVM and LSTM performed best, with LSTM using BERT embeddings achieving the highest accuracy and F1-score due to its superior ability to model sequential dependencies and leverage context-sensitive representations. Conversely, CNN, particularly when combined with FastText embeddings, exhibited the worst performance, highlighting the limitations of convolutional models in sentiment analysis tasks. The use of one-hot encoding for emoticons, while beneficial for incorporating multimodal features, remains a relatively simple approach that may not fully exploit the sentiment-bearing capacity of emoticons. Future work could explore more advanced fusion techniques, such as attention-based multimodal integration, to further enhance sentiment classification performance. These results underscore the importance of both feature representation and model selection in multimodal sentiment classification. As sentiment analysis continues to evolve, future research should focus on hybrid models that combine transformers with sequential learning architectures, as well as more advanced methods for encoding emoticons, to achieve even greater classification accuracy in multimodal sentiment analysis tasks.

*Why can AUC be high while accuracy and F1-Score are low?*

It is entirely possible for a classification model to exhibit low accuracy and F1-score while maintaining a high AUC. This occurs due to fundamental differences in how these metrics evaluate model performance. Accuracy and F1-score depend on a fixed classification threshold, typically set at 0.5, which determines how predicted probabilities are converted into discrete class labels. In contrast, AUC measures the model's ability to rank positive and negative instances correctly, independent of any specific threshold. As a result, a model can be highly effective at distinguishing

between classes in terms of probability ranking but still perform poorly when forced to classify instances at a fixed threshold.

A key reason for this phenomenon is that AUC is computed from the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate (TPR, or recall) against the false positive rate (FPR) at various classification thresholds. This means that even if a model is poorly calibrated—meaning its predicted probabilities do not translate well into accurate classifications at a chosen threshold—it can still achieve a high AUC as long as it correctly ranks most positive instances higher than negative instances. If a model assigns a slightly higher probability to a positive class compared to a negative one, it contributes positively to the AUC calculation, even if the absolute probability values are not useful for classification at a fixed threshold.

Another contributing factor is class imbalance, which can skew accuracy and F1-score but have little impact on AUC. If a dataset contains a dominant majority class and a scarce minority class, a model that predicts the majority class most of the time may yield high accuracy but poor recall for the minority class, causing a low F1-score. However, if the model still ranks positive instances above negative ones in terms of probability, the AUC will remain high. This explains why models trained on imbalanced datasets often exhibit low F1-score but high AUC—they fail to balance precision and recall but still perform well in ranking instances correctly.

Additionally, poor threshold calibration can lead to discrepancies between these metrics. If a model's predicted probabilities are systematically too high or too low, the threshold-based classification results may be misleading. For instance, if a model tends to output probabilities between 0.4 and 0.6 instead of more distinct values near 0 or 1, many predictions may fall on the wrong side of a threshold like 0.5, reducing accuracy and F1-score. However, because AUC considers the relative ranking of predictions rather than their absolute values, it remains largely unaffected by this calibration issue.

Furthermore, models that overfit to training data or struggle with high-dimensional feature spaces may exhibit a similar pattern. If a classifier is too rigid, it may fail to generalize well, leading to misclassifications at a given threshold while still preserving good overall ranking ability. This is often seen in high-dimensional text classification problems, where models learn overly specific patterns that do not translate well to new data, reducing accuracy and F1-score but keeping AUC high.

A practical example of this can be observed in SVM models with linear kernels trained on text embeddings. These models often excel at separating classes in high-dimensional space, leading to strong ranking performance and high AUC. However, if the classification threshold is not well-calibrated, they may misclassify borderline cases, leading to low accuracy and F1-score. Similarly, deep learning models such as CNNs for text classification can struggle with contextual understanding, producing uncertain probability estimates that hurt threshold-based classification while maintaining strong ranking ability.

Ultimately, the discrepancy between AUC and accuracy/F1-score highlights the need for careful threshold selection, especially in imbalanced datasets. One way to mitigate this issue is by adjusting the classification threshold based on precision-recall

trade-offs, rather than relying on a default 0.5 threshold. Another approach is to use probability calibration techniques such as Platt scaling or isotonic regression, which adjust predicted probabilities to align better with observed class distributions. In summary, a model's ability to rank positive instances correctly, even if it fails at precise classification, explains why AUC can remain high while accuracy and F1-score remain low.

### 4.2.2 Evaluation Results from 10-Fold Cross-Validation for Multiclass-based Sentiment Classification using the Second Dataset

This section presents the experimental results of the evaluation conducted using 10-fold cross-validation for multiclass sentiment classification on the second dataset. The results can be presented as Table 4.4.

Table 4.4 Evaluation Results from 10-Fold Cross-Validation for Multiclass-based Sentiment Classification using the Second Dataset

| Models | Representation for Text | Representation for Emoticon | Accuracy | F1-score | AUC |
|---|---|---|---|---|---|
| Random Forest | Word2Vec | One hot encoding | 0.9487 | 0.9482 | 0.9917 |
| | GloVE | One hot encoding | 0.9474 | 0.9468 | 0.9912 |
| | FastText | One hot encoding | 0.9661 | 0.9661 | 0.9952 |
| | Ada-002 | One hot encoding | 0.7500 | 0.7511 | 0.9117 |
| | BERT Embedding | One hot encoding | 0.9703 | 0.9700 | 0.9962 |
| SVM with Linear | Word2Vec | One hot encoding | 0.7885 | 0.7681 | 0.9768 |
| | GloVE | One hot encoding | 0.7874 | 0.7659 | 0.9732 |
| | FastText | One hot encoding | 0.7765 | 0.7525 | 0.9678 |
| | Ada-002 | One hot encoding | 0.9550 | 0.9539 | 0.9988 |
| | BERT Embedding | One hot encoding | 0.9583 | 0.9580 | 0.9691 |
| CNN | Word2Vec | One hot encoding | 0.9522 | 0.9515 | 0.9883 |
| | GloVE | One hot encoding | 0.9525 | 0.9518 | 0.9901 |
| | FastText | One hot encoding | 0.9506 | 0.9495 | 0.9916 |
| | Ada-002 | One hot encoding | 0.7670 | 0.7124 | 0.7432 |
| | BERT Embedding | One hot encoding | 0.9716 | 0.9712 | 0.9988 |
| LSTM | Word2Vec | One hot encoding | 0.9667 | 0.9661 | 0.9986 |
| | GloVE | One hot encoding | 0.9610 | 0.9603 | 0.9977 |
| | FastText | One hot encoding | 0.9418 | 0.9405 | 0.9834 |
| | Ada-002 | One hot encoding | 0.9536 | 0.9343 | 0.9856 |
| | BERT Embedding | One hot encoding | 0.9809 | 0.9809 | 0.9997 |

In Table 4.4, it can be seen that text representation plays a crucial role in the overall performance of sentiment classification models. The study utilized five different text representations: Word2Vec, GloVe, FastText, Ada-002, and BERT embeddings. Among these, BERT consistently outperformed the other methods across all models, achieving the highest accuracy, F1-score, and AUC values. This superior performance can be attributed to BERT's deep bidirectional contextual representations, which capture nuanced semantic relationships between words, making it more effective in understanding sentiment nuances.

On the other hand, Ada-002, an embedding derived from OpenAI's Ada model, exhibited the lowest performance in most cases. Its significantly lower accuracy, F1-score, and AUC suggest that it might not be well-suited for sentiment

classification tasks in this specific dataset. This can be attributed to the nature of Ada-002 embeddings, which may not be optimized for sentiment-related features, unlike domain-specific embeddings such as BERT.

The inclusion of emoticons via one-hot encoding provided an additional layer of sentiment representation, enhancing the models' ability to capture emotional context. The results indicate that the contribution of emoticons was beneficial across all models, as evidenced by the consistently high AUC scores, reflecting strong discrimination between positive and negative sentiment classes. Since emoticons inherently carry sentiment information, their structured encoding through one-hot representation allowed models to leverage non-textual features, thereby improving classification performance. However, their impact varied across different models, suggesting that some algorithms were more effective in integrating multimodal information than others.

Among the models evaluated, deep learning architectures, particularly LSTM and CNN, achieved the highest performance, demonstrating their ability to capture complex sentiment patterns. LSTM, when combined with BERT embeddings, emerged as the best-performing model, achieving an accuracy of 0.9809, an F1-score of 0.9809, and an AUC of 0.9997. This result aligns with the fundamental advantages of LSTM in handling sequential dependencies, allowing it to capture contextual relationships over long sequences effectively. The integration of BERT embeddings further enhanced LSTM's ability to understand deep semantic meanings, making it the most effective model in this study.

CNN also performed well, particularly when paired with BERT embeddings, achieving an accuracy of 0.9716 and an AUC of 0.9988. The strong performance of CNN can be attributed to its ability to extract high-level features through convolutional filters, capturing sentiment-related patterns within localized word sequences. However, CNN slightly underperformed compared to LSTM, likely due to its inability to fully capture long-term dependencies, which are crucial in understanding complex sentiment nuances.

In contrast, classical machine learning models, including Random Forest and SVM with a linear kernel, exhibited varying performance depending on the text representation method. Random Forest showed strong results with BERT embeddings, achieving an accuracy of 0.9703 and an AUC of 0.9962, indicating its effectiveness in leveraging high-dimensional features. Notably, Random Forest with FastText embeddings also performed exceptionally well, achieving an accuracy of 0.9661, suggesting that FastText's subword information helped capture meaningful representations. However, when using Ada-002 embeddings, Random Forest's accuracy dropped to 0.7500, reinforcing the notion that Ada-002 embeddings are less effective for this task.

SVM with a linear kernel demonstrated the weakest performance among the models tested, particularly when using Word2Vec, GloVe, and FastText. The accuracy ranged from 0.7765 to 0.7885, with the lowest AUC values compared to other models. This is expected, as linear classifiers struggle with high-dimensional and complex feature representations, especially when dealing with multimodal data. The performance of SVM improved significantly when paired with Ada-002 embeddings,

achieving an accuracy of 0.9550, which is an anomaly compared to its performance with other embeddings. This suggests that Ada-002 representations, despite their overall weaker performance, may have contained linearly separable features beneficial for SVM.

Overall, the study highlights the importance of selecting an appropriate combination of text representation and classification model for sentiment analysis. BERT embeddings consistently outperformed other text representations, reinforcing their state-of-the-art effectiveness in NLP tasks. LSTM emerged as the best-performing model, excelling in its ability to capture sequential dependencies and long-range contextual relationships. Conversely, SVM with linear kernels demonstrated the weakest performance, highlighting the limitations of linear models in handling complex and multimodal sentiment classification tasks.

These findings suggest that deep learning models, particularly LSTM and CNN, are more effective in sentiment classification when leveraging rich text representations like BERT. The integration of emoticons through one-hot encoding provided additional sentiment cues, improving the overall classification performance. Future research could explore advanced multimodal fusion techniques, such as attention mechanisms, to further enhance sentiment analysis capabilities.

## 4.3 Summary of the Results

The evaluation of binary-based and multiclass-based sentiment classification for multimodal movie reviews reveals key insights into model performance, text representation techniques, and the impact of multimodal fusion.

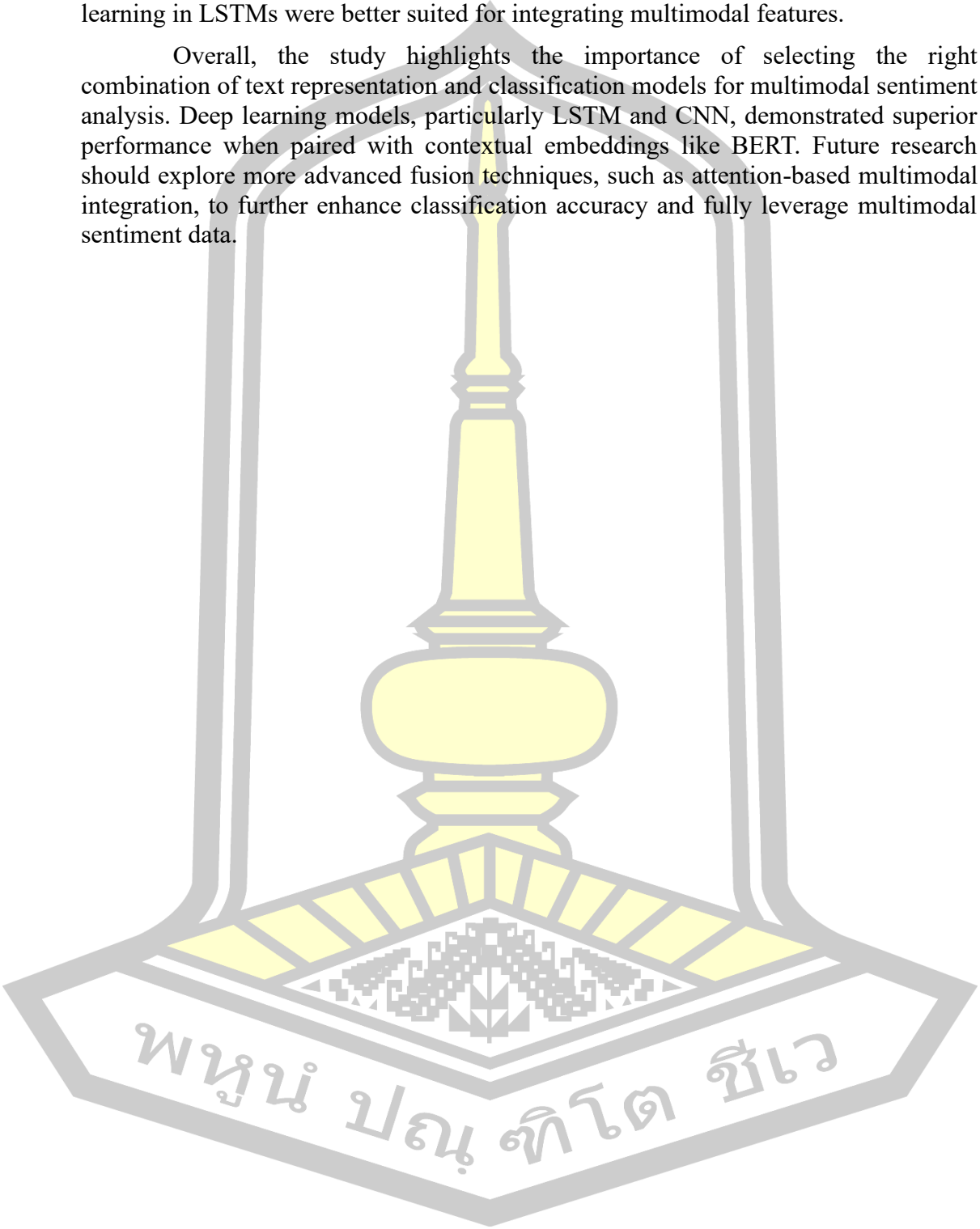
For binary sentiment classification, deep learning models, particularly LSTM and CNN, consistently outperformed traditional machine learning models, with LSTM leveraging BERT embeddings achieving the highest accuracy. The ability of LSTM to model sequential dependencies enabled it to capture sentiment nuances effectively, while CNN's hierarchical feature extraction also contributed to strong performance. Among machine learning models, Random Forest with BERT embeddings demonstrated the highest accuracy, benefiting from ensemble learning and contextualized word representations. In contrast, SVM with GloVe performed the worst, highlighting the limitations of linear classifiers when using static word embeddings.

In multiclass sentiment classification, the findings reinforced the advantages of transformer-based embeddings, with BERT and Ada-002 consistently surpassing traditional methods like Word2Vec, GloVe, and FastText. LSTM with BERT emerged as the best-performing model, demonstrating its ability to capture complex sentiment transitions across multiple classes. CNN, when paired with FastText, exhibited the lowest performance, suggesting its difficulty in modeling long-range dependencies crucial for sentiment classification. Among machine learning models, SVM with BERT embeddings performed well, while Random Forest showed variability in effectiveness depending on the embedding method used.

Across both classification tasks, the inclusion of one-hot encoded emoticons improved model performance by introducing explicit sentiment indicators. However,

its effectiveness was more pronounced in deep learning models than in machine learning models, suggesting that structured feature extraction in CNNs and sequential learning in LSTMs were better suited for integrating multimodal features.

Overall, the study highlights the importance of selecting the right combination of text representation and classification models for multimodal sentiment analysis. Deep learning models, particularly LSTM and CNN, demonstrated superior performance when paired with contextual embeddings like BERT. Future research should explore more advanced fusion techniques, such as attention-based multimodal integration, to further enhance classification accuracy and fully leverage multimodal sentiment data.



# CHAPTER 5 CONCLUSION AND FUTURE WORK

With the rapid development of the digital age , consumers are increasingly sharing their feelings through various online platforms after booking and experiencing businesses . These feedbacks are not only presented in traditional text form, but also include emoticons. Text can elaborate on consumers' evaluation of businesses, while emoticons strengthen emotional expression in an intuitive and vivid way. This paper combines consumer text and emoticons for multimodal emotion recognition, which provides a powerful tool for the consumer and other industries to deeply understand consumer needs and improve service quality.

## 5.1 Summary of Research

This study explored sentiment classification in multimodal consumer reviews by integrating textual and emoticon representations. Various text representation techniques, including Word2Vec, GloVe, FastText, Ada-002, and BERT embeddings, were examined across both machine learning and deep learning models to assess their effectiveness in binary and multiclass sentiment classification tasks. The findings reveal the crucial role of contextual embeddings in improving sentiment classification accuracy, with transformer-based embeddings such as BERT and Ada-002 consistently outperforming traditional word embeddings.

In the evaluation of machine learning models, Random Forest with BERT embeddings achieved the highest accuracy, demonstrating the advantage of ensemble learning when paired with strong contextual representations. SVM with GloVe embeddings, on the other hand, yielded the lowest performance, highlighting the limitations of linear classifiers when using static word embeddings that do not account for word meaning variations across different contexts. These results emphasize that machine learning models perform best when equipped with deep contextual embeddings, which allow for better sentiment distinction.

Deep learning models exhibited stronger performance overall, with LSTM and CNN outperforming traditional machine learning approaches. LSTM with BERT embeddings emerged as the best-performing model due to its ability to model long-range dependencies and capture contextual sentiment variations effectively. The sequential nature of LSTM allows it to retain meaning over a sequence of words, making it particularly well-suited for sentiment classification tasks that require an understanding of contextual dependencies. In contrast, CNN models, while effective in extracting local features, struggled to capture long-range dependencies, particularly when paired with embeddings such as FastText. Although CNN performed well when combined with BERT, its reliance on convolutional filters limits its ability to process highly contextualized text as effectively as LSTM.

A key contribution of this study was the integration of one-hot encoded emoticons into sentiment classification models, introducing an additional layer of sentiment representation. The results indicate that incorporating emoticons positively influenced classification performance, as emoticons serve as explicit sentiment indicators that complement textual expressions. However, the impact of emoticons varied across models. Deep learning models, particularly LSTM and CNN, leveraged

emoticon data more effectively, as their structured feature extraction mechanisms allowed for better multimodal integration. In contrast, machine learning models exhibited more limited improvements, as one-hot encoding treats emoticons as independent categorical features without capturing their semantic relationships.

Comparing the performance of text representation techniques, BERT and Ada-002 embeddings consistently outperformed Word2Vec, GloVe, and FastText. The superior performance of BERT can be attributed to its bidirectional training, which enables it to capture deep semantic and syntactic relationships in text. Similarly, Ada-002, a transformer-based embedding model, demonstrated strong performance across multiple classification tasks, suggesting that its embeddings provide a meaningful representation of sentiment-laden text. Conversely, traditional embeddings such as Word2Vec and GloVe exhibited lower performance, particularly in deep learning models, as their static nature prevents them from adapting to varying sentiment contexts.

The study also highlights the impact of data fusion techniques in multimodal sentiment classification. The chosen concatenation-based fusion approach, which combines text and emoticon representations, proved to be effective in enhancing classification accuracy. However, alternative fusion techniques, such as attention-based fusion mechanisms or transformer-based multimodal integration, could further improve sentiment classification by better capturing the interplay between textual and non-textual sentiment cues.

An interesting observation from the evaluation was the discrepancy between AUC and accuracy/F1-score in some models. Some models exhibited high AUC but relatively low accuracy and F1-score, indicating strong ranking performance but suboptimal threshold-based classification. This suggests that certain models were effective in distinguishing between positive and negative instances but struggled with precise classification at a fixed threshold. Such discrepancies highlight the importance of calibrating classification thresholds to optimize sentiment classification performance.

Overall, the findings underscore the importance of selecting appropriate text representations and classification architectures for sentiment analysis in multimodal contexts. The study demonstrates that deep learning models, particularly LSTM and CNN, achieve superior performance when paired with contextual embeddings like BERT and Ada-002. Additionally, the inclusion of emoticons through one-hot encoding enhances classification accuracy, though its impact is more pronounced in deep learning models than in machine learning models.

In practical applications, these insights suggest that businesses and researchers can enhance sentiment analysis models by leveraging transformer-based embeddings, deep learning architectures, and multimodal fusion techniques. By integrating both textual and non-textual sentiment cues, organizations can gain a more comprehensive understanding of consumer sentiment, leading to better decision-making, improved customer engagement, and enhanced brand reputation management.

## 5.2 Challenges and Limitations of This Study

Despite the promising results, this study encounters several challenges and limitations that could impact the effectiveness and generalizability of multimodal sentiment classification. The key challenges include:

1. *Difficulty in Multimodal Data Fusion* - Integrating multiple modalities (text and emoticons) presents a significant challenge due to the semantic disparity between different data types. Unlike text, which follows grammatical structures and sequential dependencies, emoticons lack explicit syntactic organization and may convey varying meanings depending on their context. The simple concatenation-based fusion method used in this study might not fully capture the complex relationships between text and emoticons. More advanced attention-based fusion mechanisms or transformer-based multimodal models could be explored to better align textual and non-textual features.
2. *Dataset Limitations and Generalization Issues* - The dataset used in this study consists of Chinese-language movie reviews from Douban, which may introduce linguistic and domain-specific biases. Since sentiment expression can vary across languages and cultural contexts, the model's performance may not generalize well to other languages or domains (e.g., product reviews, social media comments, or news articles). Additionally, the dataset size is relatively small, limiting the model's ability to learn diverse sentiment patterns. Future research could address this by incorporating larger, multilingual datasets to enhance robustness and cross-linguistic adaptability.
3. *Computational Complexity and Resource Constraints* - Deep learning models, particularly BERT, Ada-002, CNN, and LSTM, require high computational resources for training and inference. The fine-tuning of transformer-based models like BERT is computationally expensive, making it challenging for real-time applications or deployment on resource-limited devices. Additionally, hyperparameter tuning using Grid Search increases computational overhead, further limiting the feasibility of deploying these models in practical settings. Exploring lighter models like DistilBERT or efficient multimodal architectures could help mitigate these constraints while maintaining classification accuracy.

## 5.3 Future Work

Future research should explore more advanced multimodal fusion techniques, such as attention-based sentiment classification models, which allow for a more dynamic interaction between text and emoticons. Additionally, pretrained multimodal transformers, such as CLIP (Contrastive Language-Image Pretraining) or multimodal BERT models, could be investigated to further enhance multimodal sentiment analysis. Another promising direction is the use of domain-adaptive

sentiment classification, where pretrained models are fine-tuned on domain-specific consumer reviews to improve classification accuracy.

Furthermore, future studies should consider the impact of dataset size and diversity on model generalization. The datasets used in this study were derived from Chinese-language movie reviews, which may introduce language-specific characteristics. Extending the research to multilingual datasets could provide valuable insights into cross-linguistic sentiment classification and the robustness of different text representation techniques.

Finally, examining real-world deployment challenges, such as computational efficiency, scalability, and real-time processing, would be beneficial for applying multimodal sentiment analysis models in industry settings. As sentiment classification continues to evolve, combining transformer-based embeddings with innovative multimodal fusion strategies will play a key role in advancing state-of-the-art sentiment analysis methods and improving the understanding of consumer sentiment in digital media.

## REFERENCES


[1] J. Wang, Y. Hu, and J. Xiong, "The internet use, social networks, and entrepreneurship: evidence from China," *Technology Analysis & Strategic Management,* vol. 36, no. 1, pp. 122-136, 2024 2024.

[2] Z. Xiang, Q. Du, Y. Ma, and W. Fan, "A comparative analysis of major online review platforms: Implications for social media analytics in hospitality and tourism," *Tourism Management,* vol. 58, pp. 51-65, 2017 2017.

[3] K. Naithani and Y. P. Raiwani, "Realization of natural language processing and machine learning approaches for text-based sentiment analysis," *Expert Systems,* vol. 40, no. 5, p. e13114, 2023 2023.

[4] Y. Liu, X. Ding, M. Chi, J. Wu, and L. Ma, "Assessing the helpfulness of hotel reviews for information overload: a multi-view spatial feature approach," *Information Technology & Tourism,* vol. 26, no. 1, pp. 59-87, 2024 2024.

[5] S. Poria, E. Cambria, R. Bajpai, and A. Hussain, "A review of affective computing: From unimodal analysis to multimodal fusion," *Information fusion,* vol. 37, pp. 98-125, 2017 2017.

[6] J. Zhang, Z. Yin, P. Chen, and S. Nichele, "Emotion recognition using multi-modal data and machine learning techniques: A tutorial and review," *Information fusion,* vol. 59, pp. 103-126, 2020 2020.

[7] W. Peng, Z. Qin, Y. Hu, Y. Xie, and Y. Li, "Fado: Feedback-aware double controlling network for emotional support conversation," *Knowledge-Based Systems,* vol. 264, p. 110340, 2023 2023.

[8] C. Maple *et al.*, "The ai revolution: opportunities and challenges for the finance sector," *arXiv preprint arXiv:2308. 16538,* 2023 2023.

[9] M. Rodríguez-Ibánez, A. Casánez-Ventura, F. Castejón-Mateos, and P.-M. Cuenca-Jiménez, "A review on sentiment analysis from social media platforms," *Expert Systems with Applications,* vol. 223, p. 119862, 2023 2023.

[10] A. Balahur, J. M. Hermida, and A. Montoyo, "Detecting implicit expressions of sentiment in text based on commonsense knowledge," presented at the Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis (WASSA 2.011), 2011, 2011.

[11] Y. Dai, Z. Yan, J. Cheng, X. Duan, and G. Wang, "Analysis of multimodal data fusion from an information theory perspective," *Information Sciences,* vol. 623, pp. 164-183, 2023 2023.

[12] E. H. Park and V. C. Storey, "Emotion ontology studies: A framework for expressing feelings digitally and its application to sentiment analysis," *ACM Computing Surveys,* vol. 55, no. 9, pp. 1-38, 2023 2023.

[13] I. K. S. Al-Tameemi, M.-R. Feizi-Derakhshi, S. Pashazadeh, and M. Asadpour, "Multi-model fusion framework using deep learning for visual-textual sentiment classification," *Computers, Materials & Continua,* vol. 76, no. 2, pp. 2145-2177, 2023 2023.

[14] T. Zheng *et al.*, "Revisiting review helpfulness prediction: An advanced deep learning model with multimodal input from Yelp," *International Journal of Hospitality Management,* vol. 114, p. 103579, 2023 2023.

[15] A. Perti, A. Sinha, and A. Vidyarthi, "Cognitive hybrid deep learning-based multi-modal sentiment analysis for online product reviews," *ACM Transactions*

*on Asian and Low-Resource Language Information Processing,* vol. 23, no. 8, pp. 1-14, 2024 2024.

[16] Z. Liu, B. Zhou, D. Chu, Y. Sun, and L. Meng, "Modality translation-based multimodal sentiment analysis under uncertain missing modalities," *Information Fusion,* vol. 101, p. 101973, 2024 2024.

[17] G. Yi *et al.*, "Vlp2msa: expanding vision-language pre-training to multimodal sentiment analysis," *Knowledge-Based Systems,* vol. 283, p. 111136, 2024 2024.

[18] S. Georgieva, "Application of Artificial Intelligence and Machine Learning in the Conduct of Monetary Policy by Central Banks," *Икономически изследвания,* no. 8, pp. 177-199, 2023 2023.

[19] W. Gan, M. S. Dao, K. Zettsu, and Y. Sun, "Iot-based multimodal analysis for smart education: Current status, challenges and opportunities," presented at the Proceedings of the 3rd ACM Workshop on Intelligent Cross-Data Analysis and Retrieval, 2022, 2022.

[20] Z. Tang, Q. Xiao, Y. Qin, X. Zhou, J. T. Zhou, and K. Li, "Multi-View Interactive Representations for Multimodal Sentiment Analysis," *IEEE Transactions on Consumer Electronics,* vol. 70, no. 1, pp. 4095-4107, 2024 2024.

[21] E. T. Sivadasan, N. Mohana Sundaram, and R. Santhosh, "Stock market forecasting using deep learning with long short-term memory and gated recurrent unit," *Soft Computing,* vol. 28, no. 4, pp. 3267-3282, 2024 2024.

[22] H. A. Bouarara, "Recurrent neural network (RNN) to analyse mental behaviour in social media," *International Journal of Software Science and Computational Intelligence (IJSSCI),* vol. 13, no. 3, pp. 1-11, 2021 2021.

[23] M. A. I. Sunny, M. M. S. Maswood, and A. G. Alharbi, "Deep learning-based stock price prediction using LSTM and bi-directional LSTM model," presented at the 2020 2nd novel intelligent and leading emerging sciences conference (NILES), 2020, 2020.

[24] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," presented at the International conference on machine learning, 2015, 2015.

[25] B. Jang, M. Kim, G. Harerimana, S.-U. Kang, and J. W. Kim, "Bi-LSTM model to increase accuracy in text classification: Combining Word2vec CNN and attention mechanism," *Applied Sciences,* vol. 10, no. 17, p. 5841, 2020 2020.

[26] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," presented at the 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS), 2017, 2017.

[27] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing,* vol. 452, pp. 48-62, 2021 2021.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," presented at the Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), 2019, 2019.

[29] R. Patil, S. Boit, V. Gudivada, and J. Nandigam, "A survey of text representation and embedding techniques in nlp," *IEEE Access,* vol. 11, pp. 36120-36146, 2023 2023.

[30] Y. Sun *et al.*, "Modifying the one-hot encoding technique can enhance the adversarial robustness of the visual model for symbol recognition," *Expert Systems with Applications,* vol. 250, p. 123751, 2024 2024.
[31] W. A. Qader, M. M. Ameen, and B. I. Ahmed, "An overview of bag of words; importance, implementation, applications, and challenges," presented at the 2019 international engineering conference (IEC), 2019, 2019.
[32] F. Almeida and G. Xexéo, "Word embeddings: A survey," *arXiv preprint arXiv:1901. 09069,* 2019 2019.
[33] K. W. Church, "Word2Vec," *Natural Language Engineering,* vol. 23, no. 1, pp. 155-162, 2017 2017.
[34] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," presented at the Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, 2014.
[35] D. Valero-Carreras, J. Alcaraz, and M. Landete, "Comparing two SVM models through different metrics based on the confusion matrix," *Computers & Operations Research,* vol. 152, p. 106131, 2023 2023.
[36] W. Zhu, N. Zeng, N. Wang, and Others, "Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS implementations," *NESUG proceedings: health care and life sciences, Baltimore, Maryland,* vol. 19, p. 67, 2010 2010.
[37] A.-M. Šimundić, "Measures of diagnostic accuracy: basic definitions," *ejifcc,* vol. 19, no. 4, p. 203, 2009 2009.
[38] M. A. de Reus and M. P. van den Heuvel, "Estimating false positives and negatives in brain networks," *Neuroimage,* vol. 70, pp. 402-409, 2013 2013.
[39] Y. Zhu, "TP, TN, FP and FN Tables for different methods in different parameters," 2015 2015.
[40] Ş. K. Çorbacıoğlu and G. Aksel, "Receiver operating characteristic curve analysis in diagnostic accuracy studies: A guide to interpreting the area under the curve value," *Turkish journal of emergency medicine,* vol. 23, no. 4, pp. 195-198, 2023 2023.
[41] A. M. Carrington *et al.*, "Deep ROC analysis and AUC as balanced average accuracy, for improved classifier selection, audit and explanation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 45, no. 1, pp. 329-341, 2022 2022.
[42] H. Zhang and M. O. Shafiq, "Survey of transformers and towards ensemble learning using transformers for natural language processing," *Journal of big Data,* vol. 11, no. 1, p. 25, 2024 2024.
[43] K. Han *et al.*, "A survey on vision transformer," *IEEE transactions on pattern analysis and machine intelligence,* vol. 45, no. 1, pp. 87-110, 2022 2022.
[44] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on pattern analysis and machine intelligence,* vol. 41, no. 2, pp. 423-443, 2018 2018.
[45] E. H. Hovy, "What are sentiment, affect, and emotion? Applying the methodology of Michael Zock to sentiment analysis," in *Language production, cognition, and the Lexicon*: Springer, 2014, pp. 13-24.
[46] T. Sun, S. Wang, and S. Zhong, "Multi-granularity feature attention fusion network for image-text sentiment analysis," presented at the Computer Graphics

International Conference, 2022, 2022.
[47] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems,* vol. 30, 2017 2017.
[48] P. Zontone, G. Boato, J. Hare, P. Lewis, S. Siersdorfer, and E. Minack, "Image and collateral text in support of auto-annotation and sentiment analysis," 2010 2010.
[49] K. Ma, Z. Yu, K. Ji, and B. Yang, "Stream-based live public opinion monitoring approach with adaptive probabilistic topic model," *Soft Computing,* vol. 23, no. 16, pp. 7451-7470, 2019 2019.
[50] T. Chen, H. SalahEldeen, X. He, M.-Y. Kan, and D. Lu, "Velda: Relating an image tweet's text and images," presented at the Proceedings of the AAAI Conference on Artificial Intelligence, 2015, 2015.
[51] D. Joshi *et al.*, "Aesthetics and emotions in images," *IEEE Signal Processing Magazine,* vol. 28, no. 5, pp. 94-115, 2011 2011.
[52] D. Borth, R. Ji, T. Chen, T. Breuel, and S.-F. Chang, "Large-scale visual sentiment ontology and detectors using adjective noun pairs," presented at the Proceedings of the 21st ACM international conference on Multimedia, 2013, 2013.
[53] Q. You, J. Luo, H. Jin, and J. Yang, "Robust image sentiment analysis using progressively trained and domain transferred deep networks," presented at the Proceedings of the AAAI conference on Artificial Intelligence, 2015, 2015.
[54] A. Ortis, G. M. Farinella, and S. Battiato, "Survey on visual sentiment analysis," *IET Image Processing,* vol. 14, no. 8, pp. 1440-1456, 2020 2020.
[55] L. Zhu, Z. Zhu, C. Zhang, Y. Xu, and X. Kong, "Multimodal sentiment analysis based on fusion methods: A survey," *Information Fusion,* vol. 95, pp. 306-325, 2023 2023.
[56] Q. You, J. Luo, H. Jin, and J. Yang, "Joint visual-textual sentiment analysis with deep neural networks," presented at the Proceedings of the 23rd ACM international conference on Multimedia, 2015, 2015.
[57] X. Chen, Y. Wang, and Q. Liu, "Visual and textual sentiment analysis using deep fusion convolutional neural networks," presented at the 2017 IEEE International Conference on Image Processing (ICIP), 2017, 2017.
[58] A. Hu and S. Flaxman, "Multimodal sentiment analysis to explore the structure of emotions," presented at the proceedings of the 24th ACM SIGKDD international conference on Knowledge Discovery & Data Mining, 2018, 2018.
[59] J. Xu *et al.*, "Visual-textual sentiment classification with bi-directional multi-level attention networks," *Knowledge-Based Systems,* vol. 178, pp. 61-73, 2019 2019.
[60] J. Yu and J. Jiang, "Adapting BERT for target-oriented multimodal sentiment classification," 2019, 2019.
[61] Q.-T. Truong and H. W. Lauw, "Vistanet: Visual aspect attention network for multimodal sentiment analysis," presented at the Proceedings of the AAAI conference on artificial intelligence, 2019, 2019.
[62] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," presented at the ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, 2021.

[63] M. Katsurai and S. i. Satoh, "Image sentiment analysis using latent correlations among visual, textual, and sentiment views," presented at the 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), 2016, 2016.

[64] B. Jou, T. Chen, N. Pappas, M. Redi, M. Topkara, and S.-F. Chang, "Visual affect around the world: A large-scale multilingual visual sentiment ontology," presented at the Proceedings of the 23rd ACM international conference on Multimedia, 2015, 2015.

[65] T. Niu, S. Zhu, L. Pang, and A. El Saddik, "Sentiment analysis on multi-view social data," presented at the MultiMedia Modeling: 22nd International Conference, MMM 2016, Miami, FL, USA, January 4-6, 2016, Proceedings, Part II 22, 2016, 2016.

[66] X. Liu, "Multimodal public sentiment analysis model based on local semantic information," *Journal of Information Security Research,* vol. 5, no. 4, pp. 340-345, 2019 2019.

[67] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov, "SemEval-2016 task 4: Sentiment analysis in Twitter," *arXiv preprint arXiv:1912. 01973,* 2019 2019.

[68] N. Xu, W. Mao, and G. Chen, "Multi-interactive memory network for aspect based multimodal sentiment analysis," presented at the Proceedings of the AAAI conference on artificial intelligence, 2019, 2019.

[69] P. Prettenhofer and B. Stein, "Cross-language text classification using structural correspondence learning," presented at the Proceedings of the 48th annual meeting of the association for computational linguistics, 2010, 2010.

[70] U. Pavalanathan and J. Eisenstein, "Emoticons vs. emojis on Twitter: A causal inference approach," *arXiv preprint arXiv:1510. 08480,* 2015 2015.

[71] R. Das and T. D. Singh, "Multimodal sentiment analysis: a survey of methods, trends, and challenges," *ACM Computing Surveys,* vol. 55, no. 13s, pp. 1-38, 2023 2023.

[72] A. Gandhi, K. Adhvaryu, S. Poria, E. Cambria, and A. Hussain, "Multimodal sentiment analysis: A systematic review of history, datasets, multimodal fusion methods, applications, challenges and future directions," *Information Fusion,* vol. 91, pp. 424-444, 2023 2023.

[73] S. Lai, X. Hu, H. Xu, Z. Ren, and Z. Liu, "Multimodal sentiment analysis: A survey," *Displays,* vol. 80, p. 102563, 2023 2023.

[74] Z. Huang *et al.*, "Audio-oriented multimodal machine comprehension via dynamic inter-and intra-modality attention," presented at the Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 2021.

[75] H. Wen, S. You, and Y. Fu, "Cross-modal context-gated convolution for multi-modal sentiment analysis," *Pattern Recognition Letters,* vol. 146, pp. 252-259, 2021 2021.

[76] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective LSTMs for target-dependent sentiment classification," *arXiv preprint arXiv:1512. 01100,* 2015 2015.

[77] Z. Li *et al.*, "Multimodal sentiment analysis based on interactive transformer and soft mapping," *Wireless Communications and Mobile Computing,* vol. 2022, no. 1, p. 6243347, 2022 2022.

[78] W. Han, H. Chen, and S. Poria, "Improving multimodal fusion with hierarchical

mutual information maximization for multimodal sentiment analysis," *arXiv preprint arXiv:2109. 00412,* 2021 2021.

[79] H. Fang, P. Xiong, L. Xu, and Y. Chen, "Clip2video: Mastering video-text retrieval via image clip," *arXiv preprint arXiv:2106. 11097,* 2021 2021.

[80] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl, "Language-driven semantic segmentation," *arXiv preprint arXiv:2201. 03546,* 2022 2022.

[81] J. Xu *et al.*, "Groupvit: Semantic segmentation emerges from text supervision," presented at the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, 2022.

[82] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," *arXiv preprint arXiv:2104. 13921,* 2021 2021.

[83] L. H. Li *et al.*, "Grounded language-image pre-training," presented at the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, 2022.

[84] H. Zhang *et al.*, "Glipv2: unifying localization and vl understanding," presented at the 36th Conf. Neural Inf. Process. Syst. NeurIPS, 2022, 2022.

[85] H. Luo *et al.*, "Clip4clip: An empirical study of clip for end to end video clip retrieval," *arXiv preprint arXiv:2104. 08860,* 2021 2021.

[86] M. Wang, J. Xing, and Y. Liu, "Actionclip: A new paradigm for video action recognition," *arXiv preprint arXiv:2109. 08472,* 2021 2021.

[87] W. Kim, B. Son, and I. Kim, "Vilt: Vision-and-language transformer without convolution or region supervision," presented at the International conference on machine learning, 2021, 2021.

[88] J. Li, R. Selvaraju, A. Gotmare, S. Joty, C. Xiong, and S. C. H. Hoi, "Align before fuse: Vision and language representation learning with momentum distillation," *Advances in neural information processing systems,* vol. 34, pp. 9694-9705, 2021 2021.

[89] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," presented at the International conference on machine learning, 2022, 2022.

[90] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," *arXiv preprint arXiv:2205. 01917,* 2022 2022.

[91] G. Cai and B. Xia, "Convolutional neural networks for multimedia sentiment analysis," presented at the Natural Language Processing and Chinese Computing: 4th CCF Conference, NLPCC 2015, Nanchang, China, October 9-13, 2015, Proceedings 4, 2015, 2015.

[92] B. Liu *et al.*, "Context-aware social media user sentiment analysis," *Tsinghua Science and Technology,* vol. 25, no. 4, pp. 528-541, 2020 2020.

[93] J. Shang and S. Hamori, "Do Large Datasets or Hybrid Integrated Models Outperform Simple Ones in Predicting Commodity Prices and Foreign Exchange Rates?," *Journal of Risk and Financial Management,* vol. 16, no. 6, p. 298, 2023 2023.

[94] Q. You, J. Luo, H. Jin, and J. Yang, "Cross-modality consistent regression for joint visual-textual sentiment analysis of social multimedia," presented at the Proceedings of the Ninth ACM international conference on Web search and data

mining, 2016, 2016.
[95] Q. You, L. Cao, H. Jin, and J. Luo, "Robust visual-textual sentiment analysis: When attention meets tree-structured recursive neural networks," presented at the Proceedings of the 24th ACM international conference on Multimedia, 2016, 2016.
[96] S. Poria, E. Cambria, D. Hazarika, N. Mazumder, A. Zadeh, and L.-P. Morency, "Multi-level multiple attentions for contextual multimodal sentiment analysis," presented at the 2017 IEEE international conference on data mining (ICDM), 2017, 2017.
[97] M. Chen, S. Wang, P. P. Liang, T. Baltrušaitis, A. Zadeh, and L.-P. Morency, "Multimodal sentiment analysis with word-level fusion and reinforcement learning," presented at the Proceedings of the 19th ACM international conference on multimodal interaction, 2017, 2017.
[98] F. Huang, X. Zhang, Z. Zhao, J. Xu, and Z. Li, "Image--text sentiment analysis via deep multimodal attentive fusion," *Knowledge-Based Systems,* vol. 167, pp. 26-37, 2019 2019.
[99] C. Liu *et al.*, "Improving sentiment analysis accuracy with emoji embedding," *Journal of Safety Science and Resilience,* vol. 2, no. 4, pp. 246-252, 2021 2021.
[100] K. Ethayarajh, "How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings," *arXiv preprint arXiv:1909. 00512,* 2019 2019.

# BIOGRAPHY

| | |
|---|---|
| **NAME** | Wan Jun |
| **DATE OF BIRTH** | 3/12/1978 |
| **PLACE OF BIRTH** | Xiaoxi Town, Guang 'an District, Guang 'an City, Sichuan Province, China |
| **ADDRESS** | Building 27, Xiangtimancheng, Yongchuan District, Chongqing, China |
| **POSITION** | Building 27 |
| **PLACE OF WORK** | Chongqing City Vocational College |
| **EDUCATION** | 2003 Bachelor in Physics，China West Normal University<br>2012 Master of Software Engineering，UESTC (University of Electronic Science and Technology of China)<br>2022 Doctor of Philosopyhy in Computer Science, Mahasarakham University |
| **Research grants & awards** | Research on key technologies of software-defined heterogeneous Internet of Things |
| **Research output** | Two paper |