A Survey Report on Natural Language Processing and Its Core Technologies

Authors: Rao Jiansheng

Date: 2025-07-09T21:47:00+00:00

Abstract

Advances in technology and the development of the Internet have enabled the storage and distribution of large-scale unstructured data (such as audio, video, and natural language text). However, any such storage and distribution incurs certain costs, naturally prompting considerations for the efficient utilization of large-scale unstructured data. Natural Language Processing (NLP) is a discipline within computer science and artificial intelligence that investigates how to analyze these unstructured data. In essence, the core task of natural language processing is to represent unstructured data in a manner comprehensible to computers, enabling them to process such data using their inherent capabilities, and subsequently "translate" the computational results back into human-understandable language. The development of natural language processing relies on multiple disciplines, including linguistics and computer science. Linguistics provides definitions of linguistic structures and theories of meaning, while computer science furnishes the techniques and algorithms for processing and implementing these linguistic theories and definitions. These two fields complement each other, collectively supporting the automated understanding and generation of natural language by computers. In industry, natural language processing finds extensive application in tasks such as sentiment analysis, text classification, context-based text extraction, document summarization, and machine translation. The objective of this survey report is to conduct an in-depth investigation of natural language processing and its core technologies, and to discuss relevant cutting-edge technologies and models.

Full Text

A Survey Report on Natural Language Processing and Its Core Technologies

Rao Jiansheng
School of Computer Science, Sun Yat-sen University, Guangzhou

Abstract

The advancement of technology and the development of the Internet have made it possible to store and distribute large-scale unstructured data (such as audio, video, and natural language text). However, any action of storing and distributing data incurs certain costs, leading naturally to considerations of how to efficiently utilize such massive unstructured data. Natural Language Processing (NLP) is a computer science and artificial intelligence discipline that studies how to analyze these unstructured data. In essence, the core task of NLP is to represent unstructured data in a way that computers can understand, enabling computers to process the data using their strengths, and then "translate" the computational results back into human-understandable language. The development of NLP relies on many different disciplines, such as linguistics and computer science. Linguistics provides definitions of language structure and theories of meaning, while computer science offers the techniques and algorithms to process and implement these linguistic theories and definitions. The two complement each other, jointly supporting the automated understanding and generation of natural language by computers. In industry, NLP is widely applied to tasks such as sentiment analysis, text classification, context-based text extraction, document summarization, and machine translation. The purpose of this survey report is to conduct an in-depth exploration of NLP and its core technologies, and to discuss relevant frontier technologies and models.

Keywords: Natural Language Processing; Natural Language Components; Artificial Intelligence; Unstructured Data

As a discipline in computer science and artificial intelligence, the goal of natural language processing is to enable computers to recognize, understand, and generate human natural language. The objective of NLP is to teach computers how to comprehend natural language and communicate using it [**?**].

The field of natural language processing emerged around 1950, when linguists first attempted to use computers to process and analyze language. On January 7, 1954, an experiment conducted by Georgetown University and IBM demonstrated the potential of machine translation by translating over 60 Russian sentences into English. Subsequently, in 1957, linguist Noam Chomsky published *Syntactic Structures*, proposing generative grammar theory in an attempt to provide structured descriptions of unstructured human language—marking humanity's initial foray into natural language processing. Between 1950 and 1960, researchers began constructing rule-based systems that enabled computers to decode and analyze natural language through predefined rules. This early period of NLP was influenced by the development of compiler systems, as researchers attempted to treat natural language as a language generated by highly complex but inducible grammars, processing it through lexical, syntactic, and semantic analysis similar to compilers. However, this paradigm soon faced difficulties. On one hand, the grammar of natural language proved extremely difficult to summarize and formalize, making it challenging to accurately describe us-

ing computer-friendly context-free grammars (CFG). On the other hand, many words and sentence structures in natural language carry different meanings in different contexts, making it impossible to capture their behavior with finite rules.

In the 1970s, due to the limitations of rule-based systems, increased computational power, and the influence of information theory, researchers began adopting statistical methods in speech recognition and machine translation, including Hidden Markov Model (HMM)-based speech recognition and statistical machine translation (SMT) based on the "noisy channel" model from information theory. This shift marked the transition of NLP from rule-driven to data-driven approaches.

Between 1990 and 2000, researchers recognized the importance of machine learning in NLP and began using machine learning methods based on large-scale corpora, completely abandoning rule-based approaches. By 2010, with the development of deep learning, researchers started exploring the potential of deep neural networks for NLP tasks, training models capable of recognizing more complex language patterns. Deep neural networks were applied to complex tasks such as sentiment analysis, text classification, and context-based text extraction. In recent years, researchers have developed pre-trained models capable of performing more diverse tasks, such as GPT and BERT [**?**]. These pre-trained models, trained on massive corpora, have successively broken records on multiple NLP benchmark evaluations. Over more than half a century of development, NLP has become a discipline encompassing numerous techniques and technologies, including machine learning, statistical methods, and linguistic theories. NLP itself has become an essential component of many modern technological applications, such as search engines, recommendation systems, chatbots, and virtual assistants. NLP offers many advantages that enable confident deployment, including but not limited to: 1) improved efficiency, 2) enhanced task accuracy, 3) more natural human-computer interaction experiences, and 4) ease of use. NLP is primarily applied in domains such as text classification, context-based text extraction, sentiment analysis, and document classification [**?**]. The purpose of writing this paper is to understand the core components of NLP, the main methods employed, their advantages and disadvantages, and frontier developments. The remainder of this paper is organized as follows: Section 2 presents the core methods and components of NLP; Sections 3, 4, and 5 discuss frontier applications of NLP, covering multimodal sentiment analysis, short text classification, and transfer learning; Section 6 provides tracking of frontier models in NLP; and the final section summarizes the preceding content and presents conclusions. Relevant references are attached at the end of the paper.

2.1 Neural Networks

Neural networks are important models in deep learning that mimic the structure and working mechanisms of the human brain, solving problems by learning from large amounts of data. In natural language processing, neural networks are

widely applied to understand, generate, and transform natural language.

In simple terms, a neural network is a computational model composed of nodes (neurons) and connections (weights), typically consisting of an input layer, hidden layers, and an output layer. The input layer receives external data, hidden layers process data through multiple neurons to learn patterns, and the output layer decodes the hidden layer's processing results to generate human-understandable outputs. Neural networks adjust internal parameters (weights) through a training process that modifies weights in the opposite direction of the gradient of the error between network output and target output, thereby continuously reducing error. This training process is called backpropagation. Neural networks are widely applied in NLP due to several advantages: first, self-learning capability—neural networks can automatically learn data patterns and features from large datasets without manual intervention, unlike traditional machine learning methods that typically require manual feature extraction; second, powerful modeling capability—neural networks excel at handling nonlinear problems such as contextual semantics, ambiguity, and sentiment analysis in natural language.

The first neural network model applied to NLP was the Feed-Forward Neural Network (FNN) proposed by Bengio et al. (2001) [**?**]. The approach using FNN for NLP involved learning a joint probability function to estimate the probability of the next word given the previous n-1 words. In FNN, researchers first simultaneously learned word vectors and language probability models within the same architecture to address the "curse of dimensionality" problem in n-gram models. Experiments demonstrated that this neural network achieved 24%-38% lower perplexity on the Brown Corpus and Hansard Corpus compared to the state-of-the-art smoothed trigram models at the time. The feed-forward neural network marked the debut of neural networks in NLP, with its innovation lying in simultaneously learning word vectors and probability models, offering significant performance improvements over traditional n-gram models and laying the foundation for subsequent architectures such as RNN and Transformer.

Following the proposal of the feed-forward neural network model, researchers continued exploring more complex neural network architectures to further improve NLP task performance. Among these, the RNN-based NLP model proposed by Tomas Mikolov et al. (2010) became an important development direction after FNN due to its advantages in processing sequential data [**?**]. The core idea of RNN-based NLP models is to introduce recurrent connections that enable the network to retain and utilize previous state information when processing current inputs, thereby capturing temporal dependencies in sequences. This structure is particularly suitable for language modeling tasks because language itself has strong contextual dependencies. Compared to FNN models, RNN can handle longer contextual information, overcoming the limitations of fixed window sizes in FNN. However, traditional RNNs suffer from gradient vanishing and gradient explosion problems in practical applications, limiting their performance on long sequences. To address these issues, Long Short-Term

Memory (LSTM) networks emerged. LSTM effectively alleviates the gradient vanishing problem through gating mechanisms, enabling RNN to better capture long-distance dependencies. Overall, the application of RNN and its variant LSTM in NLP marked the transition of neural networks from static feature learning to dynamic sequence modeling, laying the foundation for the development of more complex models like Transformer.

The unified neural network architecture proposed by Ronan Collobert et al. (2011) further advanced the field of NLP [**?**]. This architecture achieved significant progress in multi-task learning, capable of simultaneously handling tasks such as part-of-speech tagging, named entity recognition, and semantic role labeling. By avoiding task-specific engineering designs and leveraging large amounts of unlabeled data to learn internal representations, this model demonstrated the powerful potential of neural networks for NLP tasks, laying the foundation for subsequent deep learning models.

The Transformer model proposed by Ashish Vaswani et al. (2017) marked a major breakthrough in NLP [**?**]. This model abandoned traditional recurrent neural network (RNN) and convolutional neural network (CNN) architectures, adopting a novel attention-based structure. The core innovation of Transformer is its "Self-Attention" mechanism, enabling the model to consider information from all positions in the input sequence when processing it, thereby capturing long-distance dependencies. The Transformer architecture mainly consists of an Encoder and a Decoder, each composed of multiple identical stacked layers. Each layer includes two primary sub-layers: multi-head self-attention and feed-forward fully connected networks. Multi-head self-attention allows the model to learn different representations of information in parallel across multiple subspaces, enhancing expressive power. The feed-forward network performs independent nonlinear transformations on each position's representation, further improving expressiveness. Compared to traditional RNN and LSTM models, Transformer offers significant advantages. First, it can process all positions in the input sequence in parallel, greatly improving training efficiency. Second, due to its self-attention mechanism, Transformer can capture long-distance dependencies, overcoming gradient vanishing and explosion problems that RNN and LSTM face when processing long sequences. The proposal of Transformer not only achieved remarkable results in machine translation but also provided a powerful architectural foundation for other NLP tasks such as text generation, question answering, and sentiment analysis. Its success lies in its flexible design and powerful expressive capabilities, making it one of the mainstream models in current NLP research and applications.

BERT (Bidirectional Encoder Representations from Transformers), proposed by Jacob Devlin et al. (2018), is a pre-trained language model built upon the encoder portion of the aforementioned Transformer architecture [**?**]. Unlike traditional unidirectional language models, BERT employs a bidirectional training strategy through the Masked Language Model (MLM) task, enabling the model to simultaneously capture information from both left and right contexts,

thereby achieving a more comprehensive understanding of language semantics. BERT's pre-training phase includes two main tasks: MLM and Next Sentence Prediction (NSP). In the MLM task, the model randomly masks some words in the input text and requires the model to predict these masked words based on context, thereby learning contextual word representations. In the NSP task, the model determines whether two sentences are adjacent, helping the model understand relationships between sentences. The proposal of BERT not only achieved breakthrough results in machine translation, question answering, and sentiment analysis but also propelled the NLP field toward the pre-training-fine-tuning paradigm. This approach allows models to be pre-trained on large-scale unlabeled corpora and then fine-tuned with small amounts of labeled data, achieving excellent performance across multiple tasks. BERT's success inspired subsequent research on a series of Transformer-based pre-trained models such as RoBERTa, ALBERT, and DistilBERT, which demonstrated stronger performance and higher efficiency in different tasks and scenarios. The proposal of BERT represents not only a milestone in NLP technology development but also provides new ideas and directions for the further advancement of artificial intelligence.

The GPT (Generative Pre-trained Transformer) series of models, proposed by OpenAI, marks significant progress in the field of NLP [**?**]. GPT-1 was released in 2018, adopting the decoder portion of the Transformer architecture for unsupervised pre-training, with fine-tuning for specific tasks. GPT-2 was launched in 2019, increasing parameters to 1.5 billion and demonstrating stronger text generation capabilities. In 2020, OpenAI released GPT-3 with 175 billion parameters, significantly enhancing the model's language understanding and generation capabilities. The core innovation of the GPT series lies in its generative pre-training approach—learning the statistical properties of language through unsupervised learning on massive text data, then adapting to specific tasks through fine-tuning with small amounts of labeled data. This approach enables GPT models to excel in various NLP tasks such as text generation, translation, and question answering. Unlike BERT and other models, GPT employs an autoregressive generation approach—generating the next word based on previous context—while BERT uses masked language modeling for bidirectional context modeling. GPT's generation approach gives it advantages in text generation tasks but may be less effective than BERT for understanding tasks.

2.2 Distributed Representations of Natural Language and Words

Natural language is the form of language used by humans to communicate thoughts, emotions, and information, such as Chinese and English. It features rich grammar, semantics, and contextual dependencies. Words are the basic units of natural language, carrying semantic information.

Traditional natural language processing methods (the rule-based methods mentioned in the introduction) typically treat words as discrete symbols, but this

representation struggles to capture semantic similarity and relationships between words. Synonym dictionaries (such as WordNet) were early tools for representing word semantic relationships. They organized words into synsets and defined relationships between them (such as synonymy, antonymy, hypernymy) to provide structured information for lexical semantic analysis. However, this rule-based dictionary approach suffers from limited coverage, difficulty in updating, and lack of contextual awareness.

Count-based methods construct distributed representations of words by analyzing co-occurrence frequencies of words in large-scale text corpora. A common approach is to build a word-word co-occurrence matrix where each element represents the number of times two words co-occur within a certain context window. Processing this matrix yields vector representations for each word that can capture semantic similarity between words. To improve the efficiency and effectiveness of count-based methods, researchers introduced dimensionality reduction techniques such as Singular Value Decomposition (SVD). SVD reduces computational complexity and removes noise information by decomposing high-dimensional co-occurrence matrices into lower-dimensional matrices, resulting in more compact and effective word vector representations. The Penn Treebank (PTB) dataset, as a standard corpus, is widely used to evaluate word vector models. Through training on the PTB dataset, SVD methods can effectively capture lexical semantic features and achieve good performance in various NLP tasks.

2.3 word2vec

Traditional count-based word vector methods, such as co-occurrence matrices and Latent Semantic Analysis (LSA), while capable of capturing relationships between words, suffer from the curse of dimensionality and low computational efficiency. To overcome these limitations, researchers introduced neural network-based inference methods that learn distributed representations of words through training neural network models. These methods can effectively represent word semantics in lower-dimensional spaces.

word2vec, proposed by Tomas Mikolov et al. (2013), is a word vector learning method whose core idea is to learn distributed representations of words through neural network models [**?**]. word2vec provides two model architectures: Continuous Bag-of-Words (CBOW) and Skip-gram. The CBOW model predicts the target word from context words, while Skip-gram predicts context words from the target word. Both models employ shallow neural network structures, with the training objective of maximizing the probability of context word occurrence to learn word vector representations.

Training Word2Vec models requires large amounts of text data. The training phase includes the following preprocessing steps: - Tokenization: splitting text into words or subwords - Removing stop words: eliminating common words with minimal semantic contribution - Building sliding windows: determining

the scope of context words, with a common window size of 5 - Generating training samples: constructing input-output pairs according to the CBOW or Skip-gram model structure

For example, given the sentence “I like natural language processing” with a window size of 2, the CBOW model’s training sample would be: Input: [“I”, “like”, “natural”, “language”]
Output: [“processing”]

The Skip-gram model’s training sample would be: Input: [“processing”]
Output: [“I”, “like”, “natural”, “language”]

[Figure 1: see original paper] Word2Vec model architecture diagram
[Figure 2: see original paper]

2.4 Accelerating word2vec

The core of the Word2Vec model is learning distributed representations of vocabulary through neural networks. In traditional neural networks, the output layer typically uses the Softmax function to compute probability distributions for each word. However, when the vocabulary is very large, computing Softmax becomes extremely costly. To address this problem, Mikolov et al. (2013) proposed two optimization strategies: Hierarchical Softmax and Negative Sampling [**?**]. In traditional multi-class classification problems, the Softmax function is used to compute probabilities for each category. However, in Word2Vec training, the goal is to distinguish a real word from multiple noise words, which is essentially a binary classification problem. For this purpose, Mikolov et al. proposed the negative sampling method, which transforms the multi-class problem into multiple binary classification problems using the Sigmoid function and cross-entropy loss function, thereby reducing computational load and improving training efficiency.

2.5 RNN

In natural language processing, language models are important tools that can estimate the probability distribution of word sequences. The core task of language models is to predict the next word based on contextual information, making them fundamental to natural language processing. In traditional probabilistic models, we describe word sequences through conditional probabilities, such as predicting target words based on context in Word2Vec. However, these statistical models have limitations in handling long-distance dependencies and complex contextual information.

To address these issues, Recurrent Neural Networks (RNN) were introduced into natural language processing. The design philosophy of RNN is to retain and update information from previous states through recursive connections, enabling the model to better capture temporal dependencies in word sequences. By introducing RNN into language models, we can leverage historical context

information for more accurate word prediction. RNN is a neural network model designed for sequential data. Unlike traditional feed-forward neural networks (FNN), RNN maintains historical information when processing inputs, making it highly suitable for time-series data or natural language text. The core idea of RNN is to use recurrent connections that allow the network to use the previous time step's output as the current time step's input, thereby "remembering" previous information. To train RNN, the Backpropagation Through Time (BPTT) algorithm is typically used, which computes gradients through backpropagation and updates weights accordingly. However, standard BPTT is prone to gradient vanishing and gradient explosion problems when processing long sequences. To address this, the Truncated BPTT method was developed, which truncates sequences to limit the time steps for backpropagation, reducing computational load and alleviating gradient problems. Additionally, RNN training often employs Minibatch Learning, which divides data into multiple small batches for training, further improving computational efficiency and model generalization.

RNNLM (Recurrent Neural Network Language Model) is a language model based on RNN that predicts the probability distribution of word sequences by training an RNN. Unlike traditional language models, RNNLM can learn relationships between words across longer context ranges, enabling it to better handle long-distance dependency problems. During RNNLM training, Maximum Likelihood Estimation (MLE) is typically used to learn model parameters. The training objective is to maximize the probability of word sequences, i.e., minimize the difference between predicted and actual word sequences. To evaluate RNNLM effectiveness, common metrics include Perplexity and Accuracy. Lower perplexity indicates better language understanding and more accurate word predictions. The advantages of RNNLM include its ability to model long-term dependencies in word sequences and its excellent performance across various NLP tasks, particularly in machine translation, text generation, and speech recognition. However, RNNLM also suffers from low training efficiency and slow convergence, motivating the development of improved models such as LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit).

2.6 Gated RNN

Although Recurrent Neural Networks (RNN) have demonstrated powerful capabilities in processing sequential data, they face severe gradient vanishing and gradient explosion problems in practical applications. The gradient vanishing problem means that during backpropagation through long sequences, gradients become increasingly smaller as the time step increases, preventing the network from effectively updating weights and thereby losing the ability to learn long-term dependencies. Gradient explosion refers to excessively large gradients causing weight updates that destabilize the model. Both issues limit the performance of traditional RNNs in long sequence learning. To address these problems, researchers proposed various improvements. For example, Truncated BPTT limits the time steps for backpropagation to prevent gradient vanishing or explosion

in long sequences. Additionally, Gradient Clipping methods effectively mitigate gradient explosion by limiting gradients when they exceed a certain threshold.

To solve the gradient vanishing problem in traditional RNNs, Sepp Hochreiter et al. (2017) proposed Long Short-Term Memory (LSTM) networks [**?**]. LSTM is an improved RNN structure that effectively maintains long-term memory during training through gating mechanisms, thereby overcoming the gradient vanishing problem in traditional RNNs. The core structure of LSTM includes three gates: Forget Gate, Input Gate, and Output Gate. The Forget Gate determines which information should be discarded from the memory cell; the Input Gate controls which new information can be stored in the memory cell; and the Output Gate determines which information to output from the memory cell. LSTM's design allows gradients to flow effectively across longer time steps, avoiding gradient vanishing problems. Through these gating mechanisms, LSTM can selectively retain and update memory information, enabling it to better capture long-span dependencies.

The introduction of LSTM not only solved RNN's gradient vanishing problem but also provided new directions for language model improvements. Multi-layer LSTM captures more complex patterns and longer dependencies by stacking multiple LSTM layers. Additionally, Dropout technology is widely applied in LSTM models to suppress overfitting and enhance model generalization. By randomly dropping some neurons during training, Dropout helps improve model robustness. Another common improvement method is weight sharing. In multi-layer LSTM networks, sharing weights can effectively reduce the number of model parameters, lower computational complexity, and promote information flow between different layers, enhancing model performance.

In recent years, RNN frontier technologies have continuously evolved, particularly through integration with other advanced architectures. For example, AWD-LSTM3-layer (ASGD Weight-Dropped LSTM) further suppresses overfitting and improves model generalization by incorporating weight dropping strategies into LSTM. AWD-LSTM3-layer combines LSTM's multi-layer structure with Dropout, achieving excellent results on the PTB dataset. Additionally, the introduction of Continuous Cache Pointer technology enables the model to more efficiently process long sequences, avoiding computational bottlenecks in traditional LSTM models when training on long sequences. This approach has achieved good performance in text generation and machine translation tasks, with the Continuous Cache Pointer technology being based on the famous Attention mechanism. With the introduction of the Attention mechanism, the performance of RNN series models has been further improved. The Attention mechanism enables the model to dynamically select relevant information in the sequence, allowing it to focus on key parts of the input at each time step, greatly improving the ability to capture long-distance dependencies. Section 2.8 will delve deeper into the application and improvement of Attention mechanisms in RNNs.

2.7 RNN-Based Text Generation

Text generation tasks typically include the following steps: first, input an initial seed text or context; then use a trained language model to predict the next most probable word; next, add the predicted word to the current text; finally, repeat the above process until complete text is generated. The key to this process lies in the model's ability to capture language's statistical patterns and contextual information to generate coherent and grammatically correct text.

The Seq2Seq (Sequence-to-Sequence) model proposed by Ilya Sutskever et al. (2014) is a typical encoder-decoder structure widely applied to text generation tasks [**?**]. Its basic principle is: the encoder maps the input sequence to a fixed-length context vector, and the decoder generates the target sequence based on this vector. The model's advantage lies in its ability to handle cases where input and output lengths differ, making it suitable for various natural language processing tasks. In practical implementation, the Seq2Seq encoder typically uses Recurrent Neural Networks (RNN) or their variants (such as LSTM, GRU) to process input sequences and generate hidden state sequences. The decoder then generates target sequences based on these hidden states. During training, cross-entropy loss functions are commonly used, with model parameters updated through backpropagation algorithms.

To improve Seq2Seq model performance, researchers proposed various enhancement methods. For example, Ilya Sutskever et al. (2014) proposed the Reverse Input strategy and Peeky strategy [**?**]. Reverse Input improves the model's ability to capture long-distance dependencies by reversing the input sequence; the Peeky strategy introduces partial target information during the decoder's initial stage to help the model converge faster. Additionally, the introduction of attention mechanisms allows the model to dynamically focus on different parts of the input sequence when generating each word, thereby improving generated text quality.

Seq2Seq models have achieved remarkable results in multiple NLP tasks. In machine translation, Seq2Seq models enable end-to-end translation processes, avoiding complex alignment and tokenization steps in traditional methods. In text summarization, Seq2Seq models can automatically generate concise summaries, reducing manual intervention. In dialogue generation, Seq2Seq models can generate natural and fluent responses based on user input, improving user experience. Additionally, Seq2Seq models are widely applied in image caption generation, speech recognition, and other domains.

[Figure 2: see original paper] Seq2Seq model architecture diagram

2.8 Attention

Traditional Seq2Seq models compress the entire input sequence into a fixed-length context vector when processing long sequences. This representation may lead to information loss, particularly in long sentences. The attention mecha-

nism proposed by Bahdanau et al. (2016) addresses this problem by allowing the decoder to dynamically focus on different parts of the input sequence when generating each output [**?**]. This mechanism enables the model to selectively focus on relevant parts of the input sequence based on current contextual information during decoding, significantly improving translation quality and model interpretability.

After introducing the Attention mechanism, the structure of Seq2Seq models changed. Specifically, at each decoding step, the decoder combines not only the previous time step's output but also a weighted sum of encoder hidden states to form a context vector. This context vector reflects the most relevant information from the input sequence for the current decoding step. In this way, the model can dynamically adjust its focus on different parts of the input sequence when generating each output, improving its ability to process long sequences and complex dependency relationships.

After introducing the Attention mechanism, Seq2Seq models achieved significant performance improvements in multiple NLP tasks. Particularly in machine translation, the Attention mechanism enables the model to better capture complex mapping relationships between source and target languages, improving translation accuracy and fluency. Additionally, the Attention mechanism enhances model interpretability by providing information about which parts of the input sequence the model focuses on when generating each output. However, despite its excellent performance in many tasks, the Attention mechanism has high computational complexity, especially when processing long sequences, which may lead to increased consumption of computational resources.

After introducing the Attention mechanism, researchers further explored various optimization strategies to improve model performance and efficiency. First, the Bidirectional RNN proposed by Bahdanau et al. (2016) was introduced into Seq2Seq models, enabling encoders to capture both forward and backward contextual information from input sequences, thereby providing richer contextual representations [**?**]. Second, the Attention layer has undergone multiple improvements, such as the Multi-Head Attention mechanism introduced by Ashish Vaswani et al. (2017) in the famous paper "Attention Is All You Need," which allows the model to learn information from multiple subspaces in parallel, enhancing expressive capability [**?**]. Additionally, the deep stacking of Seq2Seq models (Deep Seq2Seq) [**?**] and the introduction of Skip Connection [**?**] further improved model learning capability and training stability.

The successful application of the Attention mechanism has spawned multiple advanced models based on this mechanism. Google's Neural Machine Translation (GNMT) system (Figure 3) employs multi-layer bidirectional RNNs and Attention mechanisms, significantly improving translation quality [**?**]. Subsequently, the proposal of the Transformer model, which is entirely based on Attention mechanisms and abandons traditional RNN structures, further improved training efficiency and performance [**?**]. Additionally, models such as Neural Turing Machines (NTM) have explored more complex structures and tasks based on

the Attention mechanism, driving continuous development in the NLP field.

[Figure 3: see original paper] Google GNMT system model architecture diagram

3.1 Introduction

Sentiment analysis, sometimes also called opinion mining, is a field that studies individuals' viewpoints or emotions toward specific scenarios, topics, persons, or entities. Sentiment analysis can be divided into two types of tasks: opinion mining and emotion mining. The term "sentiment analysis" may have been first proposed by Nasukawa et al. (2003) [**?**], while the term "opinion mining" may have been first proposed by Dave et al. (2003) [**?**]. Sentiment analysis, including opinion mining, primarily analyzes or infers people's opinions expressed through their emotions, including both positive and negative emotions. Opinion holder, opinion target, and opinion are three important concepts in sentiment analysis. For example, in the text "Xiao Wang likes his phone very much," "Xiao Wang" is the opinion holder, "Xiao Wang's phone" is the opinion target, and "likes" is the opinion. The field of sentiment analysis attracted significant attention from researchers early on, and over time, it has been widely applied in industry, including government governance, enterprise management, biomedical engineering, and recommendation systems. As it falls within the scope of natural language processing research, some researchers also refer to sentiment analysis as "mini natural language processing" (Liu et al. [**?**]).

Early research in sentiment analysis primarily relied on textual data. However, with the rise of social media and short video platforms, end-users have become more diverse in how they express emotions, evolving from single text data to multimedia data including speech intonation, facial expressions, and body language. Single text analysis methods not only fail to fully capture the multidimensional characteristics of emotions but also cause information from other dimensions to be lost, wasting large-scale collected data. Against this background, Louis-Philippe Morency et al. (2011) proposed multimodal sentiment analysis [**?**], aiming to fuse information from different modalities including text, audio, and video data to improve the accuracy and robustness of emotion prediction.

In recent years, three main research trends have emerged in the sentiment analysis field: 1) lexicon-based methods, 2) machine learning-based methods, and 3) deep learning-based methods [**?**]. Lexicon-based methods primarily evaluate sentiment in text using predefined sentiment lexicons (SentiWordNet, AFINN, VADER). These models assign sentiment scores to each word in the text by querying lexicons, ranging from -1 (negative) to +1 (positive). Machine learning and deep learning methods have achieved breakthrough progress in sentiment analysis benchmark evaluations. However, since these methods typically require large amounts of data and are highly context-dependent, hybrid methods based on context, syntactic features, and lexicons are also widely used in latest research. With the emergence of contextualized networks and embedding

methods like BERT, we can more efficiently compute and extract feature representations. Transformer-based networks with thousands of parameters, such as BERT, RoBERTa, and their variants, have pushed the state of the art to new heights. Training data using modern architectures has opened new directions for sentiment analysis research, such as multimodal sentiment learning, transfer learning, multilingual sentiment analysis, and multi-domain sentiment classification.

In the visual sentiment domain, Ortis et al. [**?**] conducted a comprehensive review of visual sentiment analysis, discussing main issues, advantages and disadvantages of various methods, related datasets, and techniques. They described design principles for visual sentiment analysis systems from perspectives including emotion models, dataset definitions, and feature design, and provided formal definitions of the problem considering different granularity levels and components affecting image emotions. Zhao et al. [**?**] reviewed existing methods for image sentiment analysis, identifying two main challenges: the affective gap and perceptual subjectivity. They introduced key emotion representation models widely used in Affective Image Content Analysis (AICA) and briefly described methods available for evaluation and emotion feature extraction. Ortis et al. [**?**] again reviewed image sentiment analysis, discussing current research progress and related issues, and proposing new opportunities and challenges in this field.

In the multimodal sentiment analysis domain, Soleymani et al. [**?**] conducted extensive research on multimodal sentiment analysis, reviewing latest advances including speech comments, images, video blogs, human-computer and interpersonal interactions. They discussed challenges and opportunities in this emerging field and demonstrated the importance of multimodal sentiment analysis in natural language processing. Li et al. [**?**] outlined social media topics, describing sentiment analysis and opinion mining algorithms for social multimedia. They briefly reviewed textual sentiment analysis for social media and conducted comprehensive surveys on visual and multimodal sentiment analysis, compiling existing benchmark datasets and discussing future research trends and potential directions for multimedia sentiment analysis. Singh et al. [**?**] reported issues and challenges in social media analysis closely related to sentiment analysis. Huddar et al. [**?**] elaborated on methods, issues, and challenges in multimodal sentiment analysis. Chandrasekaran et al. [**?**] conducted detailed surveys on multimodal sentiment analysis, including feature extraction algorithms, data fusion methods, and classification techniques. Kaur et al. [**?**] surveyed opportunities and limitations in multimodal sentiment analysis. Gandhi et al. [**?**] published a survey paper reviewing main taxonomies and newly released multimodal fusion architectures.

3.2.1 Multimodal Feature Fusion and Multi-Task Learning

One of the core challenges in multimodal sentiment analysis is how to effectively fuse information from different modalities (such as text, speech, and vision).

This challenge exists because each modality (text, speech, image) carries different types and forms of emotional information, making direct fusion between modalities difficult. Additionally, aligning data from different modalities in time and space is also challenging. In videos, text and speech may have time delays, and changes in facial expressions and speech may not be synchronized. How to align these modalities and fuse data based on this alignment is a complex task. Furthermore, emotional expression in different modalities is closely related to the context of each modality, making cross-modal understanding and maintenance of contextual consistency important challenges in fusion. Therefore, in recent years, multimodal feature fusion has become a key research direction for many sentiment analysis researchers.

Multimodal Multi-Loss Fusion Network (MMFN)
Wu et al. (NAACL 2024) [**?**] proposed a Multimodal Multi-Loss Fusion Network (MMFN) that improves sentiment detection performance through multi-task learning and integration of contextual information. Figure 4 shows the multimodal sentiment analysis feature fusion network structure of MMFN. First, the network includes a cross-attention encoder that adopts a mechanism similar to self-attention, but with queries from one modality and keys and values from another modality. This cross-modal interaction helps capture dependencies between different modalities, providing more comprehensive understanding. The cross-attention formula is:

$$Attention(Q_{m1}, K_{m2}, V_{m2}) = softmax(Q_{m1}K^T)$$

where Q_{m1} is the query from the text modality, and K_{m2} and V_{m2} are keys and values from the audio modality. Through cross-attention, text features f_t are used to build query q_t, audio features f_a are used to build keys k_a and values v_a, ultimately projecting features from one modality into another modality's space. Additionally, the model introduces a self-attention encoder to capture dynamic relationships within each modality. The self-attention mechanism further strengthens understanding of specific modalities by focusing on feature representations at each time step. Finally, the model uses a Pointwise Feed-Forward Network to process features at each position and further optimizes feature representations through ReLU activation functions, enhancing multimodal fusion effectiveness.

To utilize multi-loss training, researchers also set up a fully connected layer so that each modality can output individual features and fused features. This way, each modality has an independent loss function during training, while being optimized together with the fusion network's loss. The final total loss is the weighted sum of each sub-network's loss and the fusion network's loss:

$$Loss = \sum_{m \in \{a,t,f\}} \alpha_m \cdot loss\_fn(y_m, target_m)$$

Through multi-loss training, the model can not only optimize feature learning for each modality but also improve inter-modal fusion effects, particularly when each modality has independent labels, significantly enhancing performance.

Contrastive Learning-Based Framework (CLGSI)
Yang et al. (NAACL 2024) [**?**] proposed a contrastive learning-based framework (CLGSI) that introduces sentiment intensity-guided contrastive learning and a Global-Local-Fine-grained Knowledge (GLFK) fusion mechanism (Figure 5) to improve sentiment intensity prediction.

[Figure 5: see original paper] GLFK architecture diagram

The overall architecture of the CLGSI model is shown in Figure 6 [Figure 6: see original paper]. This model combines multimodal inputs from text, video, and audio, using different encoding methods for feature extraction. The text modality uses BERT for feature extraction, while video and audio modalities use pre-trained tools for initial feature extraction, followed by further processing through Transformer encoders. The encoded representations for text, video, and audio are denoted as $I_t \in \mathbb{R}^{l_t \times d_t}$, $I_v \in \mathbb{R}^{l_v \times d_v}$, and $I_a \in \mathbb{R}^{l_a \times d_a}$, where l_m represents the sequence length for each modality and d_m represents the feature dimension for the corresponding modality.

In the common feature extraction module, the main objective is to project information from different modalities into the same representation space. For the text modality, BERT's [CLS] vector is used as the common feature representation; for video and audio modalities, the last layer output of the Transformer encoder is used as the common feature representation. Common feature representations from all modalities are input into a new representation space to further enhance information fusion between modalities. In the specific feature extraction module, Global Average Pooling is used to compress feature representations for each modality, followed by nonlinear transformations for further processing, thereby enhancing each modality's feature expression capability. The cross-modal attention encoder is used to capture dependencies between different modalities. Through cross-attention mechanisms, text features and audio features are projected onto each other, promoting inter-modal information fusion. This process helps the model extract richer emotional features from multimodal data.

To further enhance model performance, CLGSI adopts a multi-loss training strategy. Each modality's output is trained through independent fully connected layers, and the fusion network's output also has its own loss function. The final total loss is the weighted sum of each modality's loss and the fusion network's loss.

3.2.2 Missing Modality and Incomplete Data Processing

In multimodal sentiment analysis, handling missing modalities and incomplete data is a significant challenge. Missing modality refers to situations in practi-

cal applications where data from certain modalities cannot be obtained due to equipment failure, transmission issues, etc., preventing the model from fully fusing multimodal information. Due to dependencies between different modalities, missing modalities can severely impact sentiment analysis. On the other hand, incomplete data typically means that modality data contains noise, occlusion, or partial loss, making it difficult for models to extract accurate emotional features. Therefore, how to handle missing modalities, fill in missing data, and reduce the impact of noise on models has become key to improving multimodal sentiment analysis model performance. Real-world data is imperfect, and the complexity of inter-modal feature differences and data inconsistencies makes this problem even more challenging. Therefore, in recent years, missing modality and incomplete data processing have become key research directions for many sentiment analysis researchers.

Language-Dominated Noise-Resistant Learning Network (LNLN)
Zhang et al. (NeurIPS 2024) [**?**] proposed a language-dominated noise-resistant learning network (LNLN) that improves model architecture robustness under missing data conditions through dominant modality correction and multimodal learning modules. This method significantly enhances sentiment analysis performance under incomplete data conditions by processing input modalities, particularly strengthening the language modality. The LNLN pipeline includes several key steps. Figure 7 shows the LNLN pipeline. In terms of multimodal input construction and missing data processing, LNLN first introduces an embedding layer to standardize the dimensions of each modality, ensuring data consistency. For missing data, random dropout strategies are applied to each modality. The text modality uses BERT for encoding, while audio and video modalities use pre-trained tools for feature extraction, followed by further processing through Transformer encoders. This approach allows the model to maintain robustness even when inputs are missing.

LNLN identifies language as the dominant modality and enhances language feature quality through a specifically designed Dominant Modality Correction (DMC) module. This module employs adversarial learning and dynamic weighted enhancement strategies to mitigate noise impact while strengthening emotional features in the language modality. Building on this, the model also combines auxiliary modalities (such as audio and video) through the Dominant Modality-based Multimodal Learning (DMML) module to achieve effective multimodal fusion and classification, further improving sentiment analysis accuracy and robustness.

To further enhance robustness, LNLN introduces a reconstruction module that reconstructs missing modality information through Transformer layers. This way, LNLN can effectively reconstruct missing data and demonstrate stronger robustness in multimodal sentiment analysis.

LNLN adopts a multi-loss training strategy, jointly optimizing adversarial training, reconstruction loss, and sentiment prediction loss. This enables the model to not only improve modality feature extraction capabilities but also enhance

sentiment analysis performance under missing data conditions.

Multimodal Prompt Learning Method (MPLMM)
Guo et al. (ACL 2024) [**?**] proposed a multimodal prompt learning method (MPLMM) that handles the impact of missing modalities on sentiment analysis and emotion recognition through unified modality cross-attention and self-distillation mechanisms. Figure 8 shows the overall architecture of this method. Given a data batch containing different missing modality scenarios, data is first input into the Missing Modality Generation Module (MMGM) to generate features for missing modalities. These features are then passed to a pre-trained backbone network and processed together with missing-signal prompts and missing-type prompts.

[Figure 8: see original paper] Overall architecture of the MPLMM method. A batch of data containing different missing modality scenarios is fed into the missing modality generation module to obtain generated features. These features are then passed to a pre-trained backbone network and processed together with missing-signal prompts and missing-type prompts.

[Figure 9: see original paper] Generation Module (MMGM)
[Figure 10: see original paper] Missing processing

In multimodal sentiment analysis, handling missing modalities is an important challenge. Guo et al. proposed a method to recover missing modality information through generative prompts, called the Missing Modality Generation Module (MMGM). Unlike traditional methods, MMGM simplifies the process of recovering missing modalities through generative prompts. Given inputs containing audio, video, and text modalities, generative prompts are represented as $P_G = (P_{Ca}, P_{Cv}, P_{Ct})$, where P_{Ca}, P_{Cv}, and P_{Ct} represent generative prompts for audio, video, and text modalities, respectively. Equations (1) and (9) describe how these generative prompts are used to recover missing modality information:

$$\hat{x}_a = f_{t\to a}[P_{Ca}, P_{v\to a}, f_{t\to a}(x_t)]$$

Here, $\hat{x}_a$ represents the generated audio modality representation, $f_{t\to a}$ denotes the convolutional block that transforms text modality to audio modality, and P_{Ca} and $P_{v\to a}$ are generative prompts for audio and information from the video modality, respectively.

When multiple modalities are missing simultaneously, such as both audio and video modalities, the generation process is as follows:

$$\hat{x}_a = f_{t\to a}[P_{Ca}, P_{v\to a}, f_{t\to a}(x_t)]$$

Through this method, MMGM can recover missing modality information using generated prompts.

In terms of processing missing-signal and missing-type prompts, MMGM designs missing-signal prompts to help the model identify which modalities are generated and which actually exist. Specifically, missing-signal prompts can be represented as:

$$\hat{x}_a = \hat{x}_a + P_{NMS}$$

where P_{NMS} is used to indicate whether the audio modality is missing, with generated signals added to modality features through P_{NMS}. Missing-type prompts further refine missing modality information, informing the model about which modality types are missing. These prompts help the model understand missing modality types, enhancing sensitivity to missing modalities and effectively improving robustness in sentiment analysis and emotion recognition. The specific workflow of missing-type prompts is shown in Figure 10.

Through these methods, MMGM effectively handles missing modality problems in multimodal sentiment analysis and, combined with missing-signal and missing-type prompts, enhances model performance.

4.1 Introduction

The rapid development of social networks has provided people with new ways to obtain information through the Internet. Compared with traditional online information sources (such as websites), social media platforms like Weibo offer users more diverse information channels on various topics including politics and entertainment. Users have discovered that during social platform usage, they can select information content based on personal interests and gradually realize that social networks are fast, efficient, and comprehensive, leading people to increasingly prefer using social networking platforms like Weibo, WeChat, and Zhihu in daily life. On social networking platforms such as Weibo, WeChat, and Zhihu, large amounts of short text are generated daily in the form of posts or comments, typically not exceeding 200 characters. For example, Weibo comments can contain up to 140 characters, and Weibo only released the feature for dynamic content exceeding 140 characters in 2024.

However, different users have different interests and preferences. Many researchers and developers face difficulties in finding data about appropriate users at appropriate times. Therefore, a method is needed to identify data most relevant to research topics and classify text data according to involved topics or other features. Consequently, text classification has become increasingly important for many problems such as sentiment analysis, personalized recommendation, and spam filtering.

Short text classification follows basically the same steps as long text classification, typically including feature extraction and classification, using features from labeled and training data to classify texts. Compared with long text classification, the main difficulties of short text classification lie in the brevity of

documents and the sparsity of feature space. The overall process of this classification is shown in Figure 11 [Figure 11: see original paper].

[Figure 11: see original paper] Short text classification processing flow

In recent years, researchers have made various attempts on blog-type short text classification problems. Lee et al. (2011) [**?**] proposed two methods for classifying popular topics on Twitter into predefined categories. The first method relies on bag-of-words models and uses multinomial Naive Bayes algorithms to classify tweets based on text content. The second method is network-based, using C5.0 Decision Tree (DT) classifiers. This approach utilizes Twitter's social network structure, such as the number of active mutually-following users participating in specified topics. Experimental results show that the second method outperforms the first in F1-score, achieving 77% versus 70%.

Irani et al. (2010) [**?**] proposed a method for classifying tweets into corresponding categories by combining tweets with their associated web pages and using the obtained information to reduce feature space, thereby improving classification performance. Fiaidhi et al. (2013) [**?**] proposed a hierarchical ensemble classification method for topics. This method combines KNN, TF-IDF, NB, and language models to classify each tweet. The test dataset was constructed using T3C to collect popular topics and annotate them. Experimental results show that the ensemble classification method achieved about 75% accuracy and outperformed other classifiers when used individually. Additionally, they used N-gram language models to improve classifier accuracy. Weissbock et al. (2013) [**?**] addressed short text problems through data expansion methods that recursively open links contained in tweets. To increase vocabulary size, tweets recursively loaded web page titles and the top 10 most frequent words from web pages for vocabulary expansion. They also evaluated three common classifiers: Support Vector Machine (SVM), Naive Bayes (NB), and Decision Tree (DT). Results showed that using external data to expand tweets could improve classification accuracy. Wang et al. (2010) [**?**] proposed a feature selection method that utilizes short text's own data content without relying on any external data. Selected features must appear in enough documents and be representative of the topic. By separating feature selection from feature vector construction, the proposed method achieved positive results on web snippet datasets using NB classifiers. However, this method requires longer time to complete classification. Cotelo et al. (2016) [**?**] proposed a hybrid classification method combining structural information and text content. Experimental results show that this hybrid method surpasses current state-of-the-art methods. Demirsoz and Ozcan et al. (2016) [**?**] used binary classification methods to detect tweets related to news articles, with tweets classified based on text similarity methods that measure the probability of tweets belonging to specific news article-related tweet collections.

4.2.1 Application of Prompt Learning and Contrastive Learning in Short Text Classification

The application of Prompt Learning and Contrastive Learning in short text classification offers significant advantages. Prompt Learning guides models to understand input short texts by designing precise prompt templates, effectively compensating for information sparsity in short texts while reducing dependence on large amounts of labeled data and improving model performance under few-shot conditions. Contrastive Learning enhances the model's ability to distinguish short texts by learning similarities and differences between texts, particularly effective in improving classification accuracy when texts are short and contextual information is limited. Combining these two approaches enables models to better capture important features in texts, improving short text classification effectiveness.

"Soft Knowledgeable Prompt-tuning" (SKP) Method
Zhu et al. (ESWA 2024) [**?**] proposed this method, which improves model classification performance by introducing prompt learning in short text classification tasks. Figure 12 shows the structure of SKP (Soft Knowledgeable Prompt-tuning). In short text classification tasks, the SKP method consists of three main components: automatic template generation, prompt construction, and short text classification. As shown in Figure 12, the core idea of SKP is to enhance short text classification performance by introducing prompt learning and contrastive learning. First, input text and masks are mapped into pre-trained language models (PLMs) such as RoBERTa, which is used in experiments for soft prompt embedding. Unlike traditional manually designed templates, the SKP method uses neural networks (such as BiLSTM) to train soft prompts, enabling the model to better learn and adapt to characteristics of short texts. Second, considering the length of short texts, the SKP method extracts concepts related to the text from knowledge bases and expands label vocabulary through multiple strategies, effectively alleviating information sparsity in short texts. Finally, the constructed soft prompts and prompt words are combined with classification models through neural networks to predict the probability of each label word, thereby achieving classification.

[Figure 12: see original paper] SKP architecture diagram

The Verbalizer is a key component in SKP (Figure 13), whose core function is to map label words to their corresponding categories. This process helps reduce the gap between text and label vocabulary space, thereby improving downstream task performance. Unlike traditional methods that directly search for relevant concepts from large-scale knowledge bases, Verbalizer employs five strategies to expand label words extracted from short texts. These strategies not only alleviate noise impact when expanding label vocabulary but also improve efficiency and reduce overall runtime.

[Figure 13: see original paper] Verbalizer diagram

Fine-tuning Method Based on SCL Loss Function

Huang et al. (ACL 2024) [**?**] proposed a fine-tuning method based on the SCL loss function, achieving significant performance improvements on multiple subtasks. In short text classification tasks, the SCL (Supervised Contrastive Learning) model enhances text representation capability through contrastive learning. The SCL model constructs positive and negative sample pairs using multi-view augmented samples, optimizing the model through contrastive learning loss functions. In input multi-view augmented samples, positive sample pairs are different augmented views of the same instance, while negative sample pairs are other instances from different categories. The model structure and process are shown in Figure 14 [Figure 14: see original paper].

[Figure 14: see original paper] Method architecture diagram

The main process of the SCL model is as follows: First, input data is processed through a feature extractor (such as RoBERTa-base) to extract features, which are then projected into a lower-dimensional space through a projection network (a linear layer). The generated projection tensors are then used to compute the loss function, which combines supervised contrastive loss and cross-entropy loss through weighting to obtain the final loss value. The specific Supervised Contrastive Loss (SCL) is defined as:

$$L_{SCL} = \sum_{i=1}^{N} \frac{-1}{|P(i)|} \sum_{j \in P(i)} \log \frac{\exp(h_i \cdot h_j / \tau)}{\sum_{k \in K(i)} \exp(h_i \cdot h_k / \tau)}$$

where h_i and h_j represent feature vectors of positive sample pairs, h_k represents feature vectors of negative sample pairs, $P(i)$ and $K(i)$ represent positive and negative sample sets respectively, and τ is the temperature parameter controlling the smoothness of the loss function. By maximizing similarity between same-category samples and minimizing similarity between different-category samples, the model can learn more discriminative text representations. The final total loss function L_{final} is a linear combination of supervised contrastive loss and cross-entropy loss:

$$L_{final} = \alpha L_{SCL} + (1 - \alpha) L_{CE}$$

where L_{CE} is the standard cross-entropy loss and $\alpha \in [0, 1]$ is a hyperparameter controlling the weight of contrastive loss versus cross-entropy loss. This structure can obtain more useful information from augmented samples of different perspectives through contrastive learning, improving model performance in short text classification tasks.

4.2.2 Application of Time-Aware Models in Short Text Classification

Time-aware models can effectively handle dynamic features that change over time in short texts. Short text data (such as tweets and comments) typically has strong timeliness, with content and sentiment potentially changing rapidly with events. Traditional short text classification models often ignore temporal factors, while time-aware models can capture patterns of text content evolution over time by introducing timestamps and other information, thereby improving classification accuracy. Additionally, time-aware models possess adaptive capabilities, adjusting model judgments according to temporal changes, which is particularly valuable in tasks such as sentiment analysis, news classification, and social media analysis. Therefore, short text classification based on time-aware models has become a hot research topic recently.

Time-Aware Short Text Classification Model Based on CNN and BiLSTM

Omar et al. (MDPI Information 2025) [**?**] proposed a novel time-aware short text classification model that combines Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory networks (BiLSTM). By introducing timestamps and synonym data augmentation, the model significantly improves classification accuracy, particularly excelling in news topic classification tasks. Figure 15 shows the structure of the Time-aware Sentence Classification Model. By introducing temporal information to enhance the performance of short text classification models, this model addresses the problem that traditional text classification models cannot adequately consider temporal factors, especially when processing rapidly changing and time-sensitive short text data. The model improves classification effectiveness through techniques such as dynamic embedding, data preprocessing and enhancement strategies, hybrid architecture, and ensemble learning, and captures temporal information in text through Temporal-Aware Word Embeddings.

The model's structure mainly includes data preprocessing and enhancement, dynamic embedding, hybrid architecture, attention mechanism, ensemble learning, and computational complexity analysis. First, data preprocessing generates clean input data from raw text through cleaning and transformation, including lowercasing, tokenization, stop word removal, and lemmatization. Next, text data is encoded through Temporal-Aware Word Embeddings to generate time-related embedding vectors for each word. The model adopts a hybrid architecture including Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory networks (BiLSTM), which respectively capture local dependencies and long-term dependencies in short texts. Attention mechanisms are introduced to enable the model to focus more on parts most relevant to classification tasks, and ensemble learning further improves model robustness and accuracy.

[Figure 15: see original paper] Time-aware sentence classification model

To handle temporal variations of words, the model introduces the Temporal-Aware Word Embeddings module shown in Figure 16, a dynamic word embedding method. In Temporal-Aware Word Embeddings, each word's embedding is not only related to the word itself but also to a specific timestamp t. This mapping function $e(w_i, t)$ adjusts each word's embedding vector according to the specific timestamp, capturing semantic drift. This is modeled through the following formula:

$$e(w_i, t) = f_{embed}(w_i, t)$$

where f_{embed} is a mapping function that maps each word w_i and its corresponding timestamp t into a d-dimensional vector space.

To further optimize word embeddings, the model explores three temporal-aware embedding aggregation techniques (such as Aggregated Embedding Fusion (ABEF) and Temporal-Weighted Embedding Fusion (TAEF)), specifically defined as:

$$e(w_i)_{agg} = e(w_i, t)$$
$$h(s_t) = Aggregate(e(w_1, t), e(w_2, t), ..., e(w_n, t))$$

In these aggregation methods, T is the number of timestamps, $e(w_i, t)$ is the embedding vector of word w_i at time t, and $h(s_t)$ is the time-aware embedding of sentence s_t.

These methods ensure that the model can capture semantic information that changes over time, and by aggregating or weighting embedding vectors at different timestamps, further enhance the perception capability of temporal changes.

Through the combination of these techniques, the time-aware sentence classification model can demonstrate higher robustness and accuracy in short text classification tasks, especially in dynamically changing data environments, effectively addressing challenges posed by semantic changes over time.

MI-DELIGHT Model

Liu et al. (AAAI 2025) [**?**] proposed a model called MI-DELIGHT that addresses semantic sparsity problems in short text classification and improves classification effectiveness by introducing multi-source information exploration and time-aware mechanisms. The model combines key technologies including data preprocessing and enhancement, dynamic embedding, hybrid architecture, and ensemble learning, aiming to process short text datasets with limited labeled samples while fully exploiting statistical, linguistic, and factual information from the text itself.

The structure of the MI-DELIGHT model (Figure 17 [Figure 17: see original paper]) mainly includes three key components: multi-source information exploration, text representation learning, and data augmentation.

In the multi-source information exploration phase, the model constructs word graphs, POS graphs, and entity graphs to extract statistical, linguistic, and factual information from text. These graphs are updated through Graph Convolutional Networks (GCN) to obtain embedding representations for word nodes, POS nodes, and entity nodes. Specifically, word graph G_w, POS graph G_p, and entity graph G_e are used to represent semantic relationships in text, with updated node embeddings obtained through graph convolution operations:

$$H_w = GCN(G_w), \quad H_p = GCN(G_p), \quad H_e = GCN(G_e)$$

Next, the text representation learning phase combines information from different graphs to obtain the global text representation Z. The formula is described as:

$$Z_\pi = P_\pi H_\pi, \quad Z_\pi = \|Z_\pi\|_2 \quad \pi \in \{w, e, p\}$$

where P_π is the TF-IDF value between text and each word or entity in the graph, and H_π is the node embedding updated by graph convolution.

[Figure 17: see original paper] The overall architecture of MI-DELIGHT. The researcher first generates augmented samples for the input texts. Then, the original corpus $D_{org} = \{d_{org}\}$ and the augmented corpus $D_{aug} = \{d_{aug}\}$ are used to construct a word graph G_w, a POS graph G_p, and an entity graph G_e, and the text embeddings Z are obtained via the text representation learning module. Finally, these embeddings are mapped through different projection heads into different hidden spaces to which ICL, CCL, and cross-entropy (CE) are applied in a certain hierarchical order. From ICL to CCL and then to CE, the task complexity keeps increasing, and the features become more abstract. Here, feature specificity represents the abstraction level of features.

In the data augmentation phase, to overcome semantic sparsity in short texts, MI-DELIGHT expands the dataset through synonym replacement techniques. Each original text d^i_{org} is augmented to obtain d^i_{aug}, thereby increasing training data diversity. This process is implemented through the following formula:

$$\tilde{w}_i = synonym(w_i), \text{ with probability } p_{aug}$$

The innovations of MI-DELIGHT are mainly reflected in the following aspects: First, in the multi-source information exploration phase, the model simultaneously captures statistical, linguistic, and factual information from text by constructing word graphs, POS graphs, and entity graphs, thereby effectively reducing semantic sparsity in short texts. Second, the model adopts a time-aware word embedding method, making each word's representation dependent not only on the word itself but also on specific timestamps. This mechanism can capture dynamic characteristics of semantics changing over time, particularly suitable for processing time-sensitive short text data such as social media.

The time-aware word embedding formula is:

$$e(w_i, t) = f_{embed}(w_i, t)$$

Finally, MI-DELIGHT introduces a hierarchical learning task structure. Unlike traditional models, the model progressively learns through Instance-level Contrastive Learning (ICL) and Cluster-level Contrastive Learning (CCL) during training, ultimately completing the classification task. This hierarchical structure effectively enhances the model's abstraction capability and helps the model better utilize data.

The final loss function of MI-DELIGHT is optimized by combining ICL loss, CCL loss, and cross-entropy loss:

$$L = L_{CE} + \eta L_{ICL} + \zeta L_{CCL}$$

Through these innovations, MI-DELIGHT can improve classification accuracy and robustness while effectively utilizing limited labeled data in short text classification tasks, particularly showing significant advantages in dynamic social media data and other application scenarios.

5.1 Introduction

Although traditional machine learning techniques have achieved great success and been successfully applied in many industrial scenarios, certain limitations still exist in some real-world situations. The ideal machine learning scenario assumes large amounts of labeled training instances with distributions identical to test data. However, in many scenarios, collecting sufficient training data is often expensive, time-consuming, or even impractical. Semi-supervised learning can alleviate this problem to some extent by relaxing the requirement for large amounts of labeled data. Typically, semi-supervised methods only need limited labeled data and utilize large amounts of unlabeled data to improve learning accuracy. However, in many cases, unlabeled data is also difficult to collect, often leading to unsatisfactory performance of traditional models.

Transfer learning is a research paradigm proposed to address these problems. The concept of transfer learning originally stems from educational psychology. According to the generalization theory of transfer proposed by psychologist C.H. Judd, learning transfer is the result of experience generalization. As long as a person can generalize their experience, transfer from one situation to another becomes possible. This theory also states that transfer requires some connection between two learning activities. Indeed, a person who has learned violin can learn piano faster than others because both are musical instruments that may share some common knowledge. Figure 18 [Figure 18: see original paper] shows some intuitive examples of transfer learning.

[Figure 18: see original paper] Intuitive examples of transfer learning

Inspired by human ability to transfer knowledge across domains, transfer learning aims to utilize knowledge from related domains (source domains) to improve learning performance in target domains and reduce the number of required labeled samples. Notably, transferred knowledge does not always positively impact new tasks. If two domains share almost nothing in common, knowledge transfer may fail. For example, learning to ride a bicycle cannot help us learn piano faster. Moreover, similarity between domains does not always benefit learning, as these similarities may sometimes mislead learning. For instance, although Spanish and French are closely related linguistically and both belong to the Romance language family, Spanish learners may encounter difficulties when learning French, such as using wrong vocabulary or verb conjugations. This is because previous successful experience in Spanish may interfere with learning word construction, usage, pronunciation, and conjugation in French. In psychology, the phenomenon where previous experience negatively impacts learning new tasks is called negative transfer. Similarly, in transfer learning, if the target learner is negatively affected by transferred knowledge, this phenomenon is also called negative transfer. Whether negative transfer occurs may depend on multiple factors, such as the correlation between source and target domains and the learner's ability to identify transferable and beneficial knowledge across domains. Reference [**?**] provides formal definitions and some analyses of negative transfer.

Roughly speaking, based on differences between domains, transfer learning can be further divided into two categories: homogeneous transfer learning and heterogeneous transfer learning [**?**]. Homogeneous transfer learning methods are developed to handle situations where domains share the same feature space. In homogeneous transfer learning, some studies assume that differences between domains only manifest in marginal distributions, thus adapting domains by correcting sample selection bias or covariate shift. However, this assumption does not hold in many cases. For example, in sentiment classification problems, the same word may have different sentiment tendencies in different domains. This phenomenon is also called contextual feature bias. To address this problem, some studies further adapt conditional distributions.

Heterogeneous transfer learning refers to the knowledge transfer process when domains have different feature spaces. In addition to distribution adaptation, heterogeneous transfer learning also requires feature space adaptation, making it more complex than homogeneous transfer learning.

5.2.1 Trustworthy Transfer Learning and Model Risk Analysis

In transfer learning for natural language processing, trustworthy transfer learning and model risk analysis have important research necessity. Traditional transfer learning assumes similar distributions between source and target domains,

but in practical applications, domain differences may lead to unreliable knowledge transfer, thereby affecting model performance and robustness. Trustworthy transfer learning aims to improve the interpretability and reliability of knowledge during transfer, ensuring effective application of source domain knowledge in target domains. Meanwhile, model risk analysis can help assess potential risks during transfer, quantify uncertainties and errors that models may face in target domains, and provide theoretical basis for model optimization and adjustment. Therefore, combining trustworthy transfer learning and model risk analysis can enhance model adaptability in new tasks and data environments, improving the reliability and effectiveness of transfer learning in NLP.

LearnRisk-TC: Text Classification Model Based on Model Risk Analysis and Trustworthy Transfer

Sun et al. (ACL 2024) [**?**] proposed a transfer learning method LearnRisk-TC based on model risk analysis, which improves transfer learning capability in text classification by analyzing model risks, addressing the shortcomings of traditional transfer learning when target domain knowledge is insufficient. This model combines trustworthy transfer learning and model risk analysis, improving cross-domain classification performance through risk feature generation, risk model construction, and risk training optimization.

$$dist(d_i, C_j) < 0.134 \wedge knn(d_i, C_j) \geq 4 \rightarrow d_i \in C_j$$

where d_i represents document text, C_j represents text category, $dist(d_i, C_j)$ is the distance between document and class center, and $knn(d_i, C_j)$ represents the number of nearest neighbors of document d_i in class C_j.

Next, in the risk model construction phase, risk features are used to build models that estimate classifier risk. The model quantifies misclassification risk using Value-at-Risk (VaR) indicators by aggregating risk feature distributions across different categories. The risk model adjusts risk feature weights and variance through optimized learning objectives, ultimately used to evaluate misclassification risk for unlabeled instances in target domains. The VaR calculation formula is:

$$VaR(d_i) = 1 - F^{-1}(1 - \theta; u_i, \delta_i^2)$$

where u_i and δ_i^2 are the mean and variance of risk features, θ is the confidence level, and F^{-1} is the inverse cumulative distribution function of class C_i's distribution.

In the risk training phase, the model trains risk models through learning from labeled data and further optimizes classifier misclassification risk using unlabeled data. By minimizing the risk loss function, LearnRisk-TC can effectively classify target datasets. The risk loss function is defined as:

$$L_{Risk}^{test}(w) = -\sum_{i=1}^{n_s}(1 - VaR^{+}(d_i))\log(g(w_{test}(x_i)))$$

where n_s is the test set size and $g(w_{test}(x_i))$ is the classifier's prediction value for text instance x_i.

[Figure 19: see original paper] LearnRisk-TC architecture diagram

From the perspective of trustworthy transfer, LearnRisk-TC improves target domain classification effects by learning knowledge differences between source and target domains. Through risk feature generation and transfer, the model can adjust target domain classifier performance based on source domain knowledge. From the perspective of model risk analysis, LearnRisk-TC can effectively capture potential misclassification risks that models may face in target tasks by quantifying risk features, enabling adaptive adjustments during training. Ultimately, by combining these methods, LearnRisk-TC can improve the reliability and effectiveness of cross-domain transfer learning.

Through innovative methods of risk feature generation and risk model training, LearnRisk-TC provides effective solutions for risk assessment and credibility assurance in transfer learning. Its integrated risk analysis approach enables the model to fully utilize source domain knowledge under limited labeled samples, thereby improving performance and reliability of target domain classification tasks.

6 Summary and Conclusions

In this survey, we have detailedly explored core technologies, frontier applications, and key domain developments and challenges in Natural Language Processing (NLP), including multimodal sentiment analysis, short text classification, and transfer learning. As an important branch of computer science and artificial intelligence, NLP has made significant progress in multiple fields such as sentiment analysis, text classification, and machine translation. Through reviewing and analyzing technologies such as neural networks, word2vec, RNN, Gated RNN, and Attention, we have gained deep understanding of their applications and evolution in NLP.

First, neural networks, particularly deep learning models such as RNN and its variants LSTM and GRU, have become core tools for processing language tasks. The introduction of Attention mechanisms, especially the proposal of the Transformer architecture, has greatly improved NLP model performance, making multi-task learning and transfer learning new research hotspots. The success of pre-trained models like BERT and GPT has further driven NLP technology development, continuously expanding its application scenarios.

In multimodal sentiment analysis, we explored how to improve emotion recognition accuracy by fusing multimodal information including text, speech, and

vision. With the rise of social media, research on multimodal sentiment analysis has become particularly important, especially in tasks such as emotion intensity and emotion recognition. The proposal of multimodal learning frameworks has made cross-modal information sharing and learning possible, improving the robustness of sentiment analysis systems.

Short text classification is an important application direction in NLP, particularly widely used in social media and online comment analysis. This paper summarized current mainstream short text classification methods, including lexicon-based methods, machine learning-based methods, and deep learning-based methods. Although short text classification presents challenges, breakthroughs have been achieved in practical applications such as sentiment analysis and spam filtering through reasonable feature selection and classification algorithms.

In the transfer learning domain, model transfer capability and cross-domain adaptability have become key issues. By analyzing basic concepts and techniques of transfer learning, we summarized its applications and challenges in NLP, particularly how to utilize source domain knowledge to improve target domain model performance under limited labeled data conditions. Trustworthy transfer learning and model risk analysis provide theoretical support and practical guidance for further development of transfer learning.

In summary, despite significant progress in NLP research, it still faces practical challenges, especially in multimodal data fusion, feature learning for short text classification, and practical applications of transfer learning. Future research can focus on solving these specific problems to further improve model efficiency, robustness, and generalization capabilities. Through continuous optimization of existing technologies, NLP applications will become more intelligent and practical, particularly in practical scenarios such as sentiment analysis, recommendation systems, and multilingual processing.

Note: Figure translations are in progress. See original paper for figures.