A PROJECT REPORT

On

MultiDL Hindi News Categorization

Submitted in partial fulfillment of the requirements for the award of the degrees

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

Submitted by:

Devdeep Sarkar (300103321033)

Guided by:

Mrs. K. Subhashini Spurjeon
(Assistant Professor)

ASPIRE TO EXCEL

BHILAI INSTITUTE OF TECHNOLOGY DURG
DEPARTMENT OF INFORMATION TECHNOLOGY

UGC Autonomous Institution

(Affiliated to CSVTU, Approved by AICTE, NBA &NAAC ACCREDIATED)

DURG– 491001, CHHATTISGARH, INDIA
www.bitdurg.ac.in

SESSION: 2024-25

A
PROJECT REPORT
On
MultiDL Hindi News Categorization

Submitted in partial fulfillment of the
requirements for the award of the degrees

of
BACHELOR OF TECHNOLOGY
in
INFORMATION TECHNOLOGY
Submitted by:
Devdeep Sarkar (300103321033)
Guided by:
Mrs. K. Subhashini Spurjeon
(Assistant Professor)

ASPIRE TO EXCEL

BHILAI INSTITUTE OF TECHNOLOGY DURG
DEPARTMENT OF INFORMATION TECHNOLOGY

UGC Autonomous Institution

(Affiliated to CSVTU, Approved by AICTE, NBA &NAAC ACCREDIATED)

DURG– 491001, CHHATTISGARH, INDIA
www.bitdurg.ac.in

SESSION: 2024-25

CANDIDATE'S DECLARATION

We hereby declare that the project entitled **"MultiDL Hindi News Categorization"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Information Technology completed under the supervision of **Mrs. K. Subhashini Spurjeon, Assistant Professor, Information Technology,** BIT DURG is an authentic work.

Further, I/we declare that I/we have not submitted this work for the award of any other degree elsewhere.

Devdeep Sarkar

Signature and name of the student(s) with date

CERTIFICATE by PROJECT Guide(s)

It is certified that the above statement made by the students is correct to the best of my/our knowledge.

Signature of BTP Guide(s) with dates and their designation

BHILAI INSTITUTE OF TECHNOLOGY DURG

DEPARTMENT OF INFORMATION TECHNOLOGY

UGC Autonomous Institution

(Affiliated to CSVTU, Approved by AICTE, NBA &NAAC ACCREDIATED)

DURG– 491001, CHHATTISGARH, INDIA

Department of Information Technology

CERTIFICATE BY THE EXAMINERS

This is to certify that the Major Project work entitled "**MultiDL Hindi News Categorization**" is carried out by **Devdeep Sarkar (300103321033)** in partial fulfillment for the award of degree of **Bachelor of Technology** in **Information Technology**, **Chhattisgarh Swami Vivekanand Technical University**, **Durg** during the academic year 2024-2025.

Prof. Dr. Ani Thomas

Internal Guide HOD

External Examiner

ACKNOWLEDGEMENTS

We wish to acknowledge with a deep sense of hearty gratitude and indebtedness to **Dr. Vikas Pandey**, Information Technology, who gave us this opportunity to experience project work & his valuable suggestion during this project have been invaluable.

We take this opportunity to voice & record our sincerest gratefulness towards our esteem Supervisor **Mrs. K. Subhashini Spurjeon** under whose able guidance the projectwork has been brought to completion.

Our heart leaps up in thankfulness for his benevolence & time to time help, valuable suggestions, constructive criticism & active interest in the successful completion of this project work.

We are also thankful to all our honorable teachers of the Information Technology Department and ourparents whose valuable support helped us and kept us motivated all through.

Devdeep Sarkar
B.Tech. VII Sem
Discipline of Information Technology
BIT DURG

Abstract

In the modern digital world, a large number of Hindi-language papers are frequently created for official websites, news portals, and the public and private sectors. Article categorization research has exploded as internet usage rises and textual data becomes more widely available. The classification of Hindi news items is the subject of this study; there hasn't been much previous research in this area. To evaluate the classifier models, the dataset was gathered from multiple online news sources. One of the most widely spoken languages in India is Hindi. In the modern digital world, a large number of Hindi-language papers are frequently created for official websites, news portals, and the public and private sectors. The classification process is carried out in two stages primary and secondary. The data are separated into binary classes using the primary classification as India and International. The splitting of the data into multi-label classes is the main goal of the secondary classification with different news labels. Machine and deep learning algorithms were used to classify the news articles. Various three-layer hybrid classifiers models were used and evaluated in terms of accuracy, precision, F1 score, and recall, achieving a better accuracy rate. The techniques performed better once the news articles increased in the data set. The results also indicate that consistency between the number of training sets and classes and the data preparation are important factors affecting the technique's effectiveness.

List of Tables

List of Figures

List of Abbreviations

CNN	Convolution Neural Network
SVM	Support Vector Machine
LR	Logistic Regression
BERT	Bidirectional Encoder Representations From Transformer
NLP	Natural Language Processing
HLM	Hybrid Learning Model
CLS	CNN-LR-SVM
SBL	SVM-BERT-LR

Table of content

CHAPTER 1
INTRODUCTION

The thesis discusses the Machine Learning and Natural Language Processing techniques to classify Hindi news articles efficiently. The purpose of the thesis is to evaluate the performance of multiclass hybrid models and to analyse a well-structured hybrid approach that appears to be best, considering its high accuracy and less losses. This chapter briefly introduces Hindi news categorization along with its significance, challenges and their potential application areas. We also discuss the motivation of this thesis. Finally, we describe the organization of the thesis and the topicsof each chapter.

1.1 OVERVIEW OF MULTIDL HINDI NEWS CATEGORIZATION

Language plays a vital role in communication. There are thousands of dialects andlanguages spoken throughout the world. Especially, Hindi underwent significant evolution over the past millennium, emerging as a prominent global language today. Its presence on online platforms is crucial, serving as a means of expression for news outlets, government bodies, and various sectors. While Unicode facilitates online reading and writing, challenges persist, notably in Information Extraction and text classification. Text classification is particularly vital due to the abundance of uncategorized data. Leveraging natural language processing and machine learning, Hindi news articles can be effectively analysed and categorized, covering diverse topics such as world affairs, sports, politics, and economy. Machine learning and deep learning algorithms were used to classify the news articles. The classifiers were used and evaluated in terms of accuracy, precision, F1 score, and recall, achieving a better accuracy rate. Various algorithms, including Support Vector Machines, Logistic Regression, the TF-IDF vectorizer, and Convolutional Neural Networks, were used to develop a hybrid model.

1.2 SIGNIFICANCE OF HINDI TEXT CLASSIFICATION

Text classification is a common NLP task used to solve business problems in various fields. The goal of text classification is to categorize or predict a class of unseen text documents, often with the help of supervised machine learning. The classification of Hindi news articles using machine learning is an essential task with several significant aspects. Here's a detailed overview:

- **Accessibility and Information Retrieval:**
    - Improves accessibility to Hindi content for users by categorizing articles into predefined categories like sports, politics, or entertainment, aiding efficient retrieval.
    - Enables better personalization of content based on user preferences.

- **Regional Language Processing:**
    - Promotes technological advancements in Indian languages, especially Hindi, which is spoken by millions, encouraging inclusivity in AI-driven solutions.
- **Automation and Scalability:**
    - Automates the manual task of categorizing large volumes of news data, saving time and resources for media organizations.
    - Facilitates handling the continuous inflow of data from various sources like websites and social media.
- **Sentiment Analysis and Opinion Mining:**
    - Useful in analyzing public sentiment on trending topics.
    - Helps policymakers and businesses understand public opinion on socio-political issues.
- **Fake News Detection:**
    - Acts as a foundational system for building fake news detection frameworks in Hindi, critical for combating misinformation in regional languages.

1.3 APPLICATION OF HINDI NEWS CATEGORIZATION

The Hindi news classification system is a versatile tool that not only improves the accessibility and management of news content but also drives innovation in media, governance, business, and social sectors. Here are detailed application areas:

1. Media and Journalism

Content Organization: Automatically organizes and classifies news articles into predefined categories (e.g., sports, politics, entertainment). Simplifies newsroom operations by reducing manual effort in sorting news stories.

Customized News Delivery: Provides personalized news feeds to users based on their reading habits and preferences. Enables news aggregation platforms to offer curated content.

Real-Time Alerts: Helps media outlets send category-specific notifications to users (e.g., breaking news in politics or sports updates).

2. Search Engines and Information Retrieval

Improved Search Results: Enhances the relevance of search results by categorizing news articles, making them easily searchable by topic.

Semantic Search: Facilitates advanced search capabilities by understanding the context and meaning of queries in Hindi.

3. Social Media Monitoring

Trend Analysis: Identifies trending topics in specific categories (e.g., elections in politics, movie releases in entertainment).

Content Moderation: Helps in filtering and categorizing news content shared on social media platforms to ensure relevance and compliance.

Sentiment Analysis: Extracts public sentiment around news topics for businesses, policymakers, and media organizations.

4. Fake News Detection and Misinformation Control

Misinformation Identification: Serves as a foundation for building fake news detection systems in Hindi by categorizing and analyzing news content.

Fact-Checking: Supports automated fact-checking processes by identifying and tagging suspicious articles for human review.

5. E-Governance and Public Awareness

Government Portals: Helps government websites and applications categorize news articles related to schemes, policies, and updates for easier access.

Public Awareness Campaigns: Enables targeted dissemination of information in specific categories like health or education to reach the Hindi-speaking population.

6. Education and Research

Language Learning Tools: Facilitates the creation of language resources and tools for learning Hindi through classified and structured content.

Academic Research: Provides a base for linguistic and social research on media trends, regional narratives, and public discourse in Hindi.

7. Advertising and Marketing

Targeted Campaigns: Enables marketers to categorize and analyze news articles for identifying suitable platforms for ads.

Content-Based Advertising: Helps deliver advertisements related to specific categories, like sports gear ads on sports news.

8. Business Intelligence

Market Analysis: Allows businesses to understand market trends in various domains by analyzing classified news data.

Competitor Analysis: Enables companies to monitor industry news about competitors in specific categories like technology or finance.

9. News Analytics

Reader Insights: Assists media companies in understanding which categories attract the most readers and tailoring their content strategy accordingly.

Performance Metrics: Tracks the popularity of specific categories over time for strategic decision-making.

10. Legal and Regulatory

Content Compliance: Assists in monitoring news for compliance with legal and regulatory requirements by categorizing and tagging sensitive content.

Hate Speech Detection: Identifies and categorizes news articles to flag potentially harmful or inflammatory content.

11. Regional and Local News Management

Hyperlocal Journalism: Supports local news categorization for better management and distribution of region-specific content.

Diversity in News Coverage: Encourages the inclusion of underrepresented topics by identifying gaps in news coverage across categories.

12. Disaster Management and Crisis Response

Real-Time Information: Helps categorize and distribute critical information during natural disasters, pandemics, or crises (e.g., health, rescue, and recovery updates).

Alerts and Warnings: Enables targeted communication of warnings to affected regions through categorized news.

1.4 CHALLENGES OF HINDI NEWS CATEGORIZATION

1. **Linguistic Complexity:**
   - Hindi has rich morphology, with numerous word forms, synonyms, and a flexible word order, complicating text analysis and feature extraction.
   - The presence of regional dialects and mixed-code texts (e.g., Hindi-English) makes preprocessing more difficult.
2. **Limited Resources:**
   - Scarcity of high-quality labeled datasets for Hindi news articles.
   - Lack of robust pre-trained language models for Hindi, though models like multilingual BERT partially address this.
3. **Vocabulary and Ambiguity:**
   - Hindi has a large vocabulary, including loanwords from Sanskrit, Urdu, and English, adding complexity to tokenization and vectorization.
   - Words may have multiple meanings depending on context.

4. **Data Imbalance:**
    - Categories like "politics" and "entertainment" might dominate over others like "technology," leading to biased models.
5. **Preprocessing Challenges:**
    - Handling non-standard writing styles, informal spellings, and typos commonly seen in user-generated news content.
    - Dealing with punctuation, stopwords, and stemming in Hindi.
6. **Evaluation Complexity:**
    - Difficulty in defining evaluation metrics that effectively capture semantic nuances in Hindi text.
7. **Computational Costs:**
    - Training deep learning models like BERT on resource-constrained systems can be challenging due to high computational and memory requirements.

By integrating this system, organizations can unlock significant operational efficiencies and better serve the Hindi-speaking population.

1.5 MOTIVATION

The motivation for Hindi news classification stems from several societal, technological, and business-driven factors:-

- Bridging the Language Gap

  Enable millions of Hindi-speaking users to access categorized and relevant news in their native language, overcoming the dominance of English-based systems. Foster technological inclusion for Hindi and other Indian languages in AI applications, promoting linguistic and cultural diversity.

- Enhancing AI Capabilities for Hindi

  Advance the field of natural language processing (NLP) for Hindi, contributing to the creation of robust language models and tools. Motivate the collection and labeling of high-quality Hindi datasets for future research and applications**.**

- Supporting Media and Journalism

  Reduce the manual effort required in categorizing and publishing news articles, allowing journalists to focus on core content creation. Personalize news delivery for readers by offering content tailored to their preferences and interests.

- Improving Information Retrieval

  Simplify the management of large volumes of Hindi news content by automating categorization into predefined topics. Make news more searchable and accessible for users, especially for specific queries like politics, sports, or entertainment.

1.6 AN OVERVIEW OF THESIS

The thesis describes different phases and issues for developing a hindi news categorization in multi classes using machine and deep learning models. A Hindi Text Classification model is proposed, which accepts a set of known Hindi documents, preprocesses them at document, sentence and word levels, extracts features, and trains hybrid classifier, which further classifies a set of Hindi unknown documents. The thesis is organized into the following chapters:

Chapter 2 gives the complete literature survey of Text categorization research work done in India and foreign countries.

Chapter 3 provides the theoretical background of the research work in which different Machine learning and deep learning models and its application areas are discussed.

Chapter 4 is related to pre-processing phase and different hybrid models used for classification of hindi news. It describes the pre-processing stage, various linguistic tools, and other methodologies.

Chapter 5 gives results and a discussion of the research work. Finally, in Chapter 6, we draw some conclusions and present suggestions for further work.

1.7 SUMMARY

This chapter introduces . It is followed by application areas of machine translation, challenges of the machine translation system, and motivation. The chapter concludes with a brief outline of the thesis. The next chapter provides a survey of the existing literature in the field of machine categorization.

CHAPTER 2
REVIEW OF LITERATURE

2.1 INTRODUCTION

Hindi news categorization is an advanced application in natural language processing that focuses on efficiently classifying Hindi news articles into predefined categories. This approach leverages a hybrid model where various techniques and models are combined to enhance efficiency and achieve highly accurate results. In this chapter, a complete survey has been conducted to identify the problem statement. The survey work is divided into two parts. In the first part, research work already conducted in the domain of text classification is reviewed, with a particular focus on existing classification systems for English articles. In the second part, gaps in the existing research are identified. This chapter is structured into two sections: Section 2.2 discusses existing classification systems for English articles, while Section 2.3 highlights the gaps in current research with respect to Hindi news categorization.

2.2 EXISTING CLASSIFICATION SYSTEM FOR ENGLISH ARTICLES

Vipin et al. proposed a hybrid CNN-LSTM and LSTM-CNN models for sentiment classification into positive, neutral, and negative categories, achieving promising results. [A]

Parthiban et al. compared classifiers like LSTM, DNN-BiLSTM, LR, CNN, RF, and RNN across seven datasets, with the DNN-BiLSTM model outperforming others, achieving good accuracy on the combined Enron dataset. [B]

Ratnam et al. proposed a hybrid model incorporating RNNs and autoencoders was designed, with RNNs capturing document sequential dependencies and autoencoders enhancing feature representation. To improve document clustering, a hybrid model combining RNNs and autoencoders was developed, leveraging the strengths of both architectures for more efficient clustering. [C]

Livieris et al. introduced a two-level ensemble meta-learning strategy that fuses baseline classifier committees. The key idea is to improve performance by enhancing classifier diversity, ensuring the ensemble captures diverse data patterns for more accurate results. According to Livieris et al., the averaging method is the most straightforward approach for combining the predictions of several models. [D][E]

Alzoubi et al. delved into text classification using five different methods and meticulously examined the results. They explored the impact of data preparation by assessing both raw and pre-processed data.

Techniques such as morphological examination, standardization, and simplification were employed on the data. The study compared the effectiveness of approaches including Naive Bayes (NB), Support Vector Machines (SVM), Long Short-Term Memory (LTSM), Logistic Regression (LR), and Random Forest (RF). LTSM emerged as the most efficient method in terms of training time and accuracy, highlighting the significance of coherence between training set size, categories, and normalized data. Additionally, the choice of feature extraction method, particularly TF-IDF, significantly influenced the outcomes, with data simplification demonstrating the most significant impact among all preparation steps. [F]

McCallum et al. discussed advanced methods for selecting important features in text classification, such as mutual information and information gain. They highlighted Naive Bayes as a popular algorithm for such tasks but noted its data sparsity issue. Yin et al. (2017) emphasized the growing prevalence of deep neural networks (DNNs), particularly Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), in natural language processing (NLP) tasks due to their strong expressive capability and reduced need for feature engineering. [G]

Umer et al. emphasized the effectiveness of Convolutional Neural Network (CNN) models in the classification of both short and long texts. They highlighted the utility of CNNs in scenarios where class distribution is balanced, owing to their ability to deliver comparable performance while maintaining computational efficiency. This finding underscores the practicality of CNNs as a reliable solution for various text classification tasks. [H]

M. Arora et al. conducted an in-depth exploration of three distinct feature engineering techniques—CountVectorizer, TF-IDF, and Word2Vec—for preprocessing data used in training machine learning algorithms. Through extensive experimentation, they determined that the combination of Multinomial Naïve Bayes with CountVectorizer offered the highest accuracy among the methods evaluated. This finding underscores the importance of choosing appropriate feature engineering techniques for optimizing machine learning model performance. [I]

Kulkarni et al. explored the potential of various deep learning models for classifying Marathi text. Their research revealed that simple, single-layer architectures based on Convolutional Neural Networks (CNN) and Long Short-Term Memory networks (LSTM), when paired with FastText embeddings, outperformed more complex models such as those leveraging BERT-based embeddings. This result highlights the efficacy of lightweight and straightforward architectures in resource-constrained or specific linguistic contexts. [J]

Khuntia, M. et al. investigated news headline classification by employing LSTM networks along with word embeddings, cosine similarity index, and BERT-based models. They observed significant variations in classification accuracy across different categories, with notable challenges in achieving high accuracy for politics and health headlines. These discrepancies were primarily attributed to insufficient training data for these specific classes, underscoring the need for balanced and comprehensive datasets in achieving consistent model performance. [K]

Jamshidi et al. proposed a novel classification framework combining BERT for word embeddings with an MTM LSTM architecture to effectively utilize both high-level text features and contextual relationships. Their findings demonstrated that integrating BERT embeddings with LSTM networks significantly enhanced classification accuracy by capturing deeper semantic and contextual relationships within the text. This approach underscores the value of combining advanced word embedding techniques with sequential modeling for robust text classification solutions. [L]

Sitaula, C. et al. introduced a hybrid feature approach for tweet classification, combining syntactic information from bag-of-words with semantic data from fastText and domain-specific methods. They proposed a multi-channel CNN (MCNN) model to capture multi-scale information, improving sentiment classification into positive, neutral, and negative classes. Domain-specific results vary depending on the parameters and features used. Since text classification relies heavily on semantic and sentiment information and dataset size, achieving consistently high accuracy across different domains using the same techniques is challenging. [M][N][O]

2.3 GAPS IN EXISTING RESEARCH

Previous research on text classification often overlooked multi-class classification, primarily focusing on single or dual combinations of machine learning and deep learning algorithms. Furthermore, most existing models are designed to classify articles into a limited number of categories, typically ranging from 3 to 5 classes. Additionally, most analyses have been conducted in English, with few systems available for Indian languages. Addressing these gaps is essential for advancing research in text classification, enabling the development of models that are versatile, inclusive, and effective for Hindi language.

2.4 CONCLUSION

Finally, after conducting a comprehensive literature survey, the following objectives have been identified for this research work:

a) To develop a robust multi-class classification model that takes Hindi news articles as input and predicts one of ten possible categories.

b) To automate the process of classifying Hindi news articles, reducing manual effort and increasing efficiency.

c) To design and implement a hybrid model for precise categorization of Hindi news articles into the identified categories.

d) To address challenges such as the inefficiency of manual segregation, the scarcity of effective models for Hindi news classification, and to establish a foundation for future advancements in this domain.

After identifying these research objectives through the literature review, their implementation is carried out by exploring advanced concepts of machine learning, hybrid model techniques, and a detailed analysis of Hindi news article structures, which is discussed in Chapter 3.

CHAPTER 3
THEORETICAL BACKGROUND

3.1 INTRODUCTION

In the previous chapter, research objectives are already derived through a literature review. The theoretical background chapter is an essential chapter of any research work.It incorporates all vital factors, parameters, and theoretical concepts required for research work that are used to implement research objectives. In computer science, machine learning (ML), natural language processing (NLP), and artificial intelligence (AI) are related but distinct technologies. Machine Learning and Natural Language Processing are important subfields of Artificial Intelligence that have gained prominence in recent times. Combined with machine learning algorithm, NLP generate system that learn to perform task on their own and get better through experience.

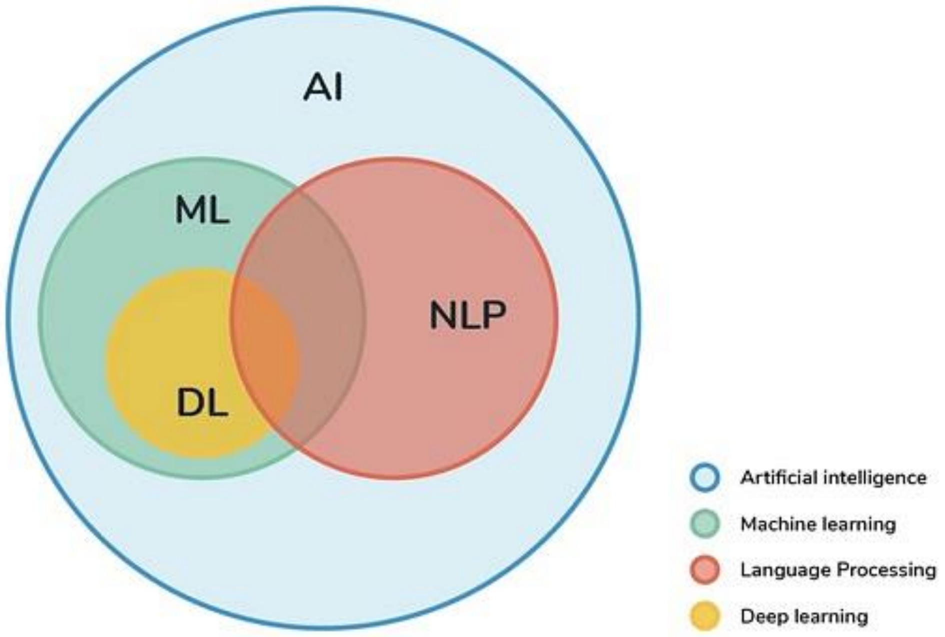


Figure 3.1 Relation among AI, ML & NLP

This chapter is organized into Section 3.2, which is about the use of NLP in the field of Computer Science; section 3.3 discusses machine translation and its different approaches;at the last Section 3.4 discusses the Linguistic background of Hindi and Chhattisgarhi languages.

3.2 NLP IN THE FIELD OF COMPUTER SCIENCE

Natural Language Processing has a subfield called computational linguistics concerned with natural language processing. The computer system converts the information from a database into human-readable form NLP deals with both speech and text. Still, speech processing is a different area to work in as it requires voice samples instead of text.

In NLP, different techniques are required to understand the commands given in human language so that the system can work according to them. To communicate with computers, we need formal languages like Java, C, C + +, Python, etc. Understanding these languages is a tedious task and requires a lot of effort. So, this is their limitation in communicating with computers. Compared to this, communication in natural language with the computer system is more straightforward. (Bharti *et al.*, 1994) discussed that Natural Language Processing is an essential subfield of artificial intelligence as the computer will be considered intelligent if it can understand commands in natural language. Natural language processing can be necessary for various applications like machine translation, text summarization, and question-answering systems. The initial processing of natural language is more straightforward, but as we go deep into processing, the difficulty increases in understanding natural language. With the increased ability of computer processing, intelligent machines are more in demand.

NLP uses machine learning to enable a machine to understand how humans communicate with one another. It also leverages datasets to create tools that understand the syntax, semantics, and the context of a particular conversation. Today, NLP powers much of the technology that we use at home and in business.

Natural Language Processing (NLP) and Machine Learning (ML) are synergistically used in text classification to transform raw text into structured and actionable insights. NLP focuses on processing and understanding human language by breaking down text into tokens, removing stop words, performing stemming or lemmatization, and converting text into numerical representations such as Term Frequency-Inverse Document Frequency (TF-IDF), Word2Vec, or advanced embeddings like BERT. These representations capture semantic and syntactic features of the text, making it suitable for ML models. Machine learning, on the other hand, applies statistical algorithms such as Logistic Regression, Support Vector Machines (SVM), or deep learning architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to identify patterns and classify the text into predefined categories. Together, NLP handles the preprocessing and feature extraction, while ML models use these features to learn and make predictions, enabling applications like sentiment analysis, spam detection, topic classification, and more. This combination bridges the gap between unstructured language data and computational decision-making, creating robust and scalable text classification systems.

3.3 BACKGROUND OF TEXT CLASSIFICATION SYSTEM

Text classification systems have evolved over decades, driven by the need to manage and analyze large volumes of textual data efficiently. From simple rule-based systems to modern deep learning techniques, the field has undergone significant transformations. Here's a detailed overview of the background:

1. Rule-Based Systems

Early text classification systems were rule-based, relying on manually crafted rules for categorizing text. These rules were typically based on keywords, patterns, or specific linguistic constructs (e.g., presence of words like "sports" or "politics").

Limitations: Highly dependent on domain expertise and manual effort. Poor scalability and adaptability to dynamic data.

2. Statistical Methods

Introduction of Probabilistic Models: Statistical approaches like Naïve Bayes emerged in the late 20th century, leveraging probabilities to predict the category of a text. These methods relied on term frequencies and conditional probabilities to classify text.

Vector Space Models: Representing text as vectors using methods like Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) allowed machines to quantify textual information.

Advantages: Automated the classification process. Scaled better than rule-based systems.

Limitations: Could not capture the semantic meaning of text. Struggled with polysemy (words with multiple meanings) and synonymy (different words with similar meanings).

3. Machine Learning Era

Supervised Learning Models: Models like Support Vector Machines (SVM), Logistic Regression, and Decision Trees became popular for text classification. These methods required labeled training data and performed well when combined with feature engineering techniques like TF-IDF or n-grams.

Feature Engineering: Techniques like n-grams, stemming, lemmatization, and stopword removal were used to preprocess text and create input features.

Advantages: Improved accuracy and robustness compared to rule-based and statistical methods.

Challenges: Heavily reliant on feature engineering. Struggled to capture deeper semantic relationships and context.

4. Rise of Embedding Techniques

Introduction of Word Embeddings: Word2Vec, GloVe, and FastText revolutionized text classification by representing words as dense, continuous vectors in a semantic space.

These embeddings captured the contextual similarity between words, enabling better classification.

Advantages:

Reduced the need for manual feature engineering. Improved handling of polysemy and synonymy.

Challenges:

Could not capture contextual variations effectively (e.g., the meaning of a word depending on the sentence).

5. Deep Learning Era

Neural Network Models: Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) brought end-to-end learning to text classification. CNNs were effective in identifying n-gram patterns, while RNNs (e.g., LSTMs, GRUs) captured sequential dependencies in text.

Pre-Trained Models: Transfer learning became a game-changer with pre-trained models like ELMo, ULMFit, and eventually, transformer-based models like BERT and GPT.

Advantages: Eliminated the need for extensive feature engineering. Achieved state-of-the-art results across various text classification tasks.

Challenges: Computationally expensive. Required large labeled datasets for fine-tuning.

6. Transformer Models and Beyond

Transformer Architecture: The introduction of the Transformer architecture, and models like BERT, RoBERTa, and GPT, brought unprecedented advances in text classification.

These models are pre-trained on massive corpora and fine-tuned on specific tasks, making them highly versatile.

Contextual Understanding: Bidirectional processing (e.g., in BERT) allowed models to understand the context of a word in a sentence more accurately.

Applications: Used for sentiment analysis, topic classification, spam detection, and more.

Challenges: High resource requirements for training and fine-tuning. Need for large labeled datasets for specialized tasks.

7. Current Trends

Multilingual Models: Models like mBERT and IndicBERT support multiple languages, enabling text classification in regional and low-resource languages.

Hybrid Models: Combining traditional machine learning methods (e.g., SVM, Logistic Regression) with embeddings from pre-trained models.

Few-Shot and Zero-Shot Learning: Emerging approaches like GPT-4 allow classification with minimal labeled data, making them suitable for low-resource scenarios.

Focus on Low-Resource Languages: Increasing emphasis on developing datasets, embeddings, and models for languages like Hindi to democratize AI adoption.

The journey of text classification systems reflects the broader evolution of artificial intelligence, from manual and rule-based methods to deep learning and transformer-based solutions. Each phase has contributed tools and techniques that address specific challenges, and today's systems are capable of understanding and categorizing text with remarkable accuracy. However, challenges like resource intensity, handling mixed languages, and low-resource scenarios continue to drive research in this field.

3.4 MACHINE LEARNING MODEL

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. Machine learning uses algorithms to analyze large amounts of data, identify patterns, and make predictions. The algorithms improve over time as they are trained with more data. The learning system of a machine learning algorithm into three main parts:

Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labeled or unlabeled, your algorithm will produce an estimate about a pattern in the data.

An Error Function: An error function evaluates the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.
A Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this iterative "evaluate and optimize" process, updating weights autonomously until a threshold of accuracy has been met.

Machine learning can be broadly classified into three types based on the nature of the learning system and the data available:
Supervised learning: In this approach, the model is trained on a labeled dataset. In other words, the data is accompanied by a label that the model is trying to predict. This could be anything from a category label to a real-valued number. The model learns a mapping between the input (features) and the output (label) during the training process. Once trained, the model can predict the output for new, unseen data. Common examples of supervised learning algorithms include for regression problems and logistic regression and support vector machines for classification problems.

Unsupervised learning: This type of learning is often used for clustering and dimensionality reduction. Clustering involves grouping similar data points together, while dimensionality reduction involves reducing the number of random variables under consideration by obtaining a set of principal variables. Common examples of unsupervised learning algorithms include k-means for clustering problems and Principal Component Analysis (PCA) for dimensionality reduction problems.

Reinforcement Learning: In reinforcement learning, the algorithm learns actions for a given set of states that lead to a goal state. It is a feedback-based learning model that takes feedback signals after each state or action by interacting with the environment. This feedback works as a reward (positive for each good action and negative for each bad action), and the agent's goal is to maximize the positive rewards to improve their performance. The behavior of the model in reinforcement learning is similar to human learning, as humans learn things by experiences as feedback and interact with the environment.

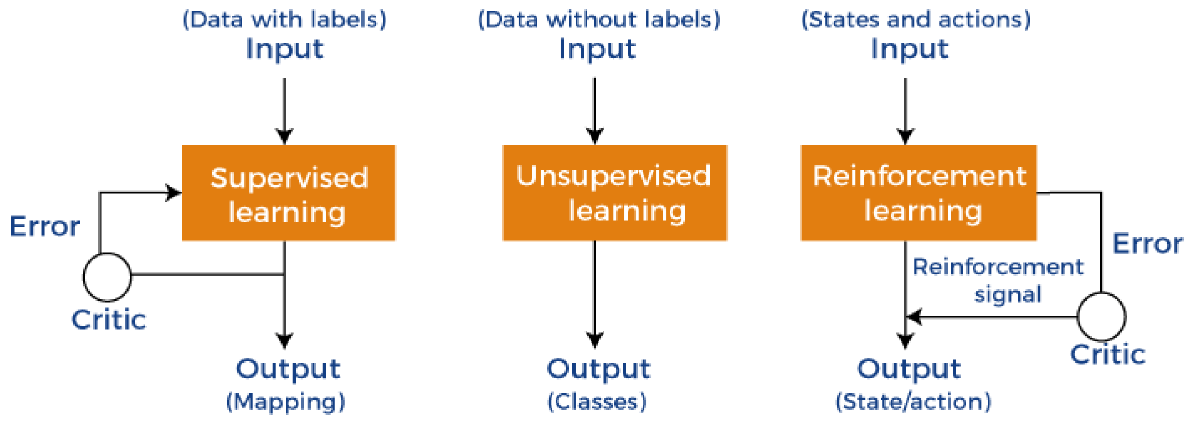


Figure 3.2 Classification of ML models

3.4.1 Support Vector Machine

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm primarily used for classification tasks. It works by identifying the optimal hyperplane that separates data points belonging to different categories with the maximum possible margin. SVMs are commonly used within classification problems. They distinguish between two classes by finding the optimal hyperplane that maximizes the margin between the closest data points of opposite classes. The number of features in the input data determine if the hyperplane is a line in a 2-D space or a plane in a n-dimensional space.

Since multiple hyperplanes can be found to differentiate classes, maximizing the margin between points enables the algorithm to find the best decision boundary between classes. This, in turn, enables it to generalize well to new data and make accurate classification predictions. The lines that are adjacent to the optimal hyperplane are known as support vectors as these vectors run through the data points that determine the maximal margin.

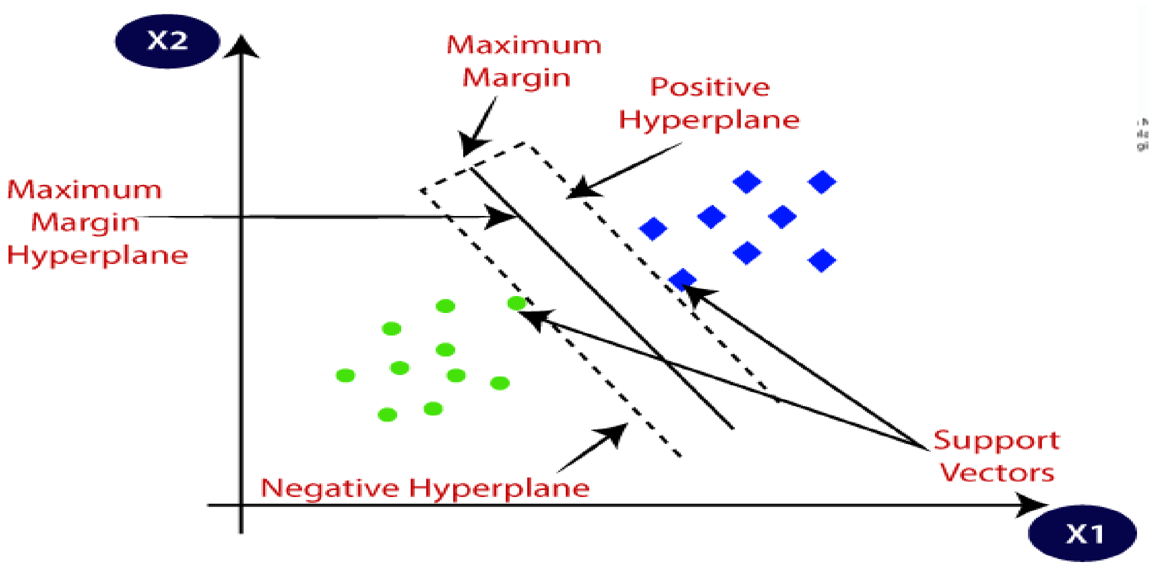


Figure 3.3 Implementation of SVM

In the context of Hindi news categorization, SVM is effective for classifying text into predefined categories such as politics, sports, or entertainment. Hindi news articles are first converted into numerical representations, using techniques like Term Frequency-Inverse Document Frequency (TF-IDF) or Word2Vec embeddings, to serve as inputs for the SVM model. The algorithm excels in handling high-dimensional data like text and is particularly useful for smaller datasets, providing robust results when the data is well-preprocessed. SVM requires numerical inputs, so Hindi text is preprocessed and converted into numerical vectors using techniques like TF-IDF or Word2Vec. Once converted, SVM classifies the articles into predefined categories (e.g., sports, politics).

3.4.2 Logistic Regression

LR is a statistical model used for classification tasks, particularly binary or multiclass problems. It predicts the probability of each category using the logistic function and assigns the label with

the highest probability. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given data set of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1.
In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. Logistic regression uses a logistic function called a sigmoid function to map predictions and their probabilities. The sigmoid function refers to an S-shaped curve that converts any real value to a range between 0 and 1. Moreover, if the output of the sigmoid function (estimated probability) is greater than a predefined threshold on the graph, the model predicts that the instance belongs to that class. If the estimated probability is less than the predefined threshold, the model predicts that the instance does not belong to the class.

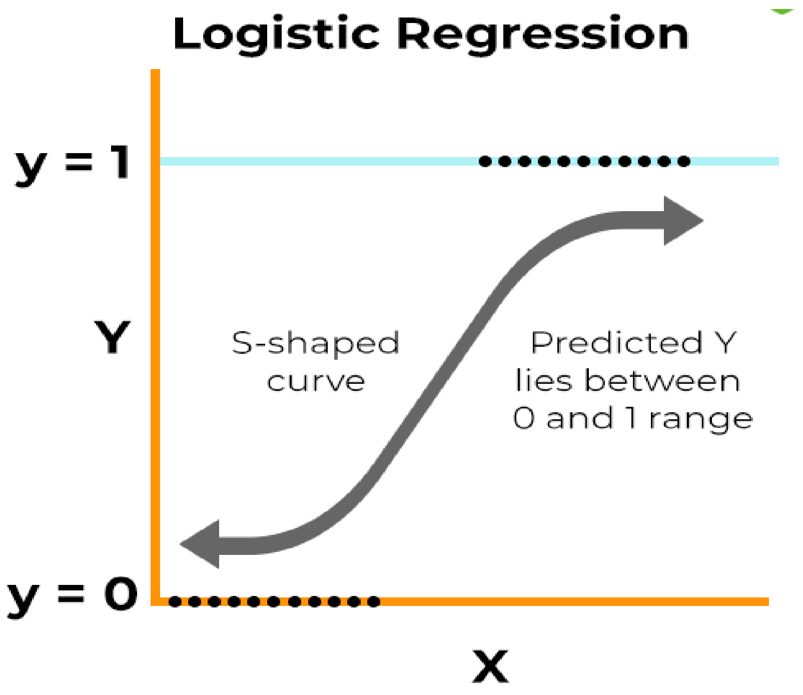


Figure 3.4 Implementation of LR

In Hindi news categorization, Logistic Regression can classify articles into various categories like entertainment, politics, or technology. Preprocessed text data, often converted into numerical forms like TF-IDF vectors or embeddings, is input into the model. Logistic Regression is simple, interpretable, and efficient for linearly separable data, making it suitable for text classification tasks when computational simplicity is a priority.

3.4.3 Bidirectional Encoder Representations from Transformers

BERT, an acronym for Bidirectional Encoder Representations from Transformers, stands as an open-source machine learning framework designed for the realm of natural language processing (NLP). Originating in 2018, this framework was crafted by researchers from Google AI Language. BERT is designed to generate a language model so, only the encoder mechanism is used. Sequence of tokens are fed to the Transformer encoder. These tokens are first embedded into vectors and then processed in the neural network. The output is a sequence of vectors, each corresponding to an input token, providing contextualized representations.

BERT is a state-of-the-art pre-trained transformer model designed to understand the context of words in a sentence by processing text bidirectionally. It provides deep contextual embeddings that capture the semantic and syntactic nuances of the text. When training language models, defining a prediction goal is a challenge. Many models predict the next word in a sequence, which is a directional approach and may limit context learning. BERT addresses this challenge with two innovative training strategies which includes Masked Language Model (MLM) and Next Sentence Prediction (NSP).

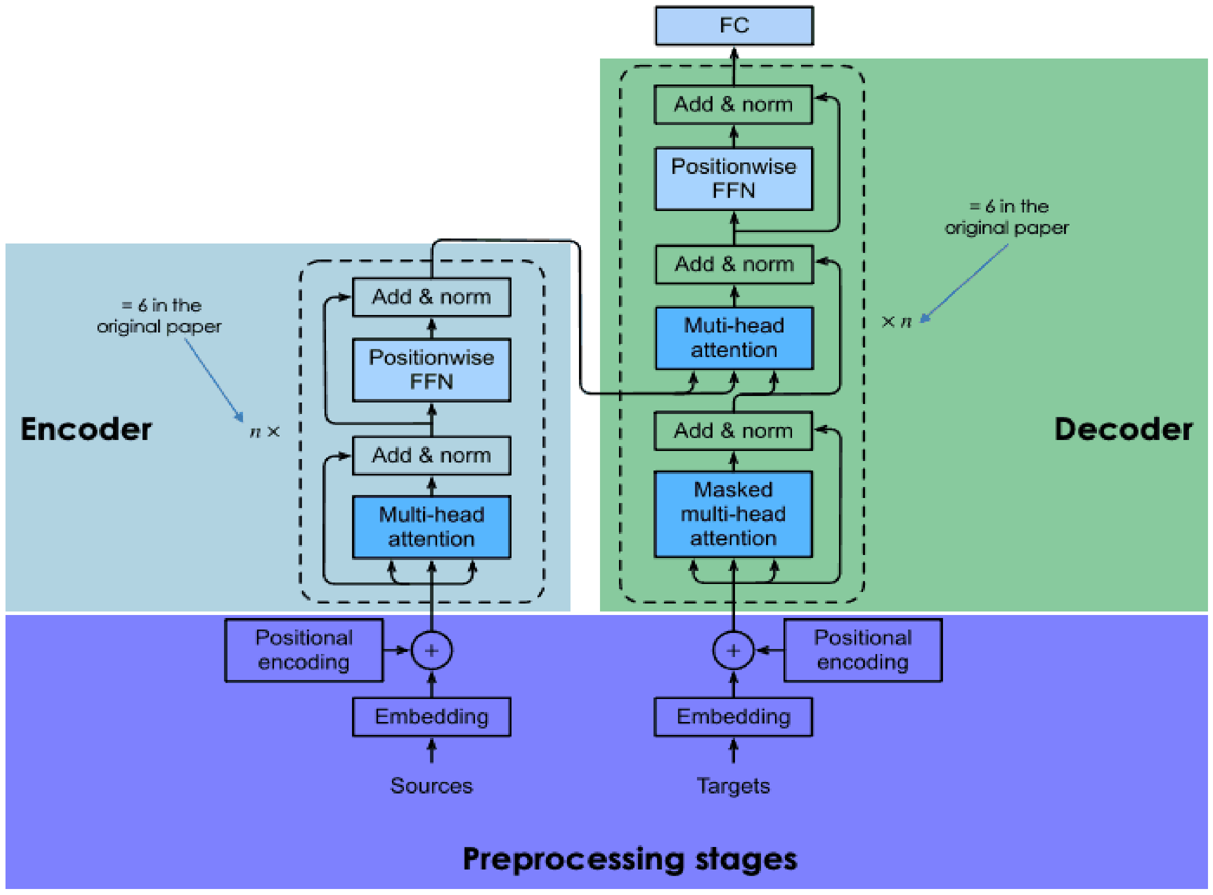


Figure 3.5 Architecture of Bert

For Hindi news categorization, BERT models like Multilingual BERT (mBERT) or IndicBERT are often used to handle the linguistic complexity of Hindi. The model can be fine-tuned on a labeled dataset of Hindi news articles, enabling it to classify articles into categories with high accuracy. BERT's ability to manage complex language structures, including Hindi's rich morphology and mixed-language text (e.g., Hinglish), makes it a robust choice for this task.

3.5 DEEP LEARNING MODEL

The definition of Deep learning is that it is the branch of machine learning that is based on artificial neural network architecture. In a fully connected Deep neural network, there is an input layer and one or more hidden layers connected one after the other. Each neuron receives input from the previous layer neurons or the input layer. The output of one neuron becomes the input to other neurons in the next layer of the network, and this process continues until the final layer produces the output of the network. The layers of the neural network transform the input data through a series of nonlinear transformations, allowing the network to learn complex representations of the input data.

Artificial neural networks are built on the principles of the structure and operation of human neurons. It is also known as neural networks or neural nets. The most widely used architectures in deep learning are feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs).

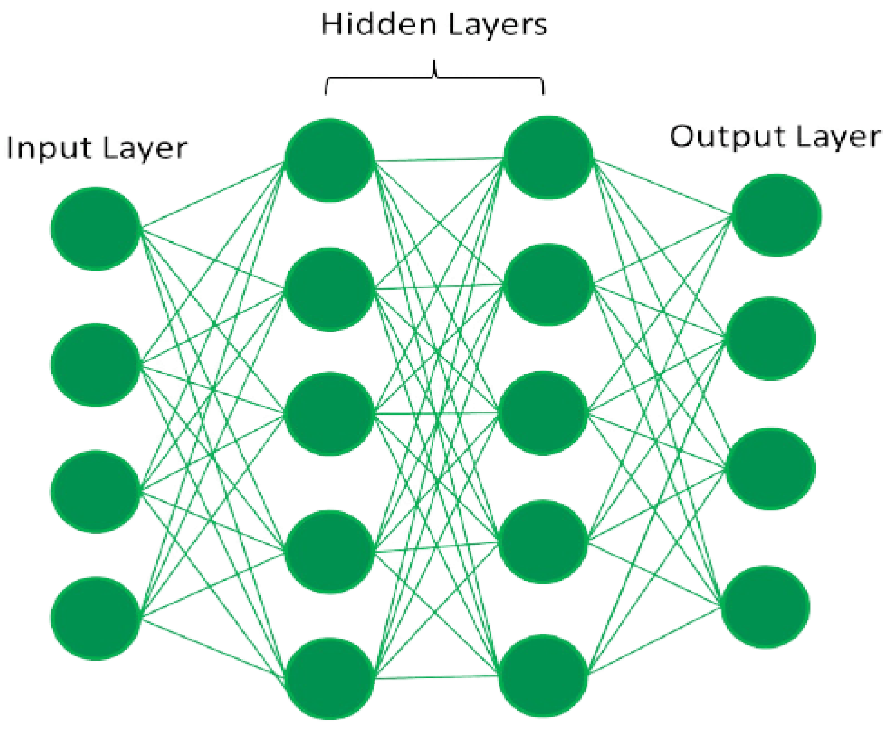


Figure 3.6 Fully Connected Artificial Neural Network

3.5.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN), also known as ConvNet, is a specialized type of deep learning algorithm mainly designed for tasks that necessitate object recognition, including image classification, detection, and segmentation. CNN originally designed for image processing, are also effective for text classification. They extract features from text by learning patterns through convolutional filters, making them adept at capturing local dependencies and n-gram relationships. CNNs are employed in a variety of practical scenarios, such as autonomous vehicles, security camera systems, and others. Some parts of CNN are:
Convolutional layer: This layer uses filters, also known as kernels, to convolve with input images to capture relevant features and patterns. The convolutional layer learns to detect edges, textures, shapes, and other visual elements.
Pooling layer: This layer reduces the dimension of the feature map by applying aggregation operations. This reduces the memory used while training the network.
Fully connected layer: This layer connects all inputs into a designated set of output neurons. It typically corresponds to the last layer of a CNN.
Output layer: This is the final layer of the CNN and produces the network's predictions. The number of neurons in this layer depends on the task.

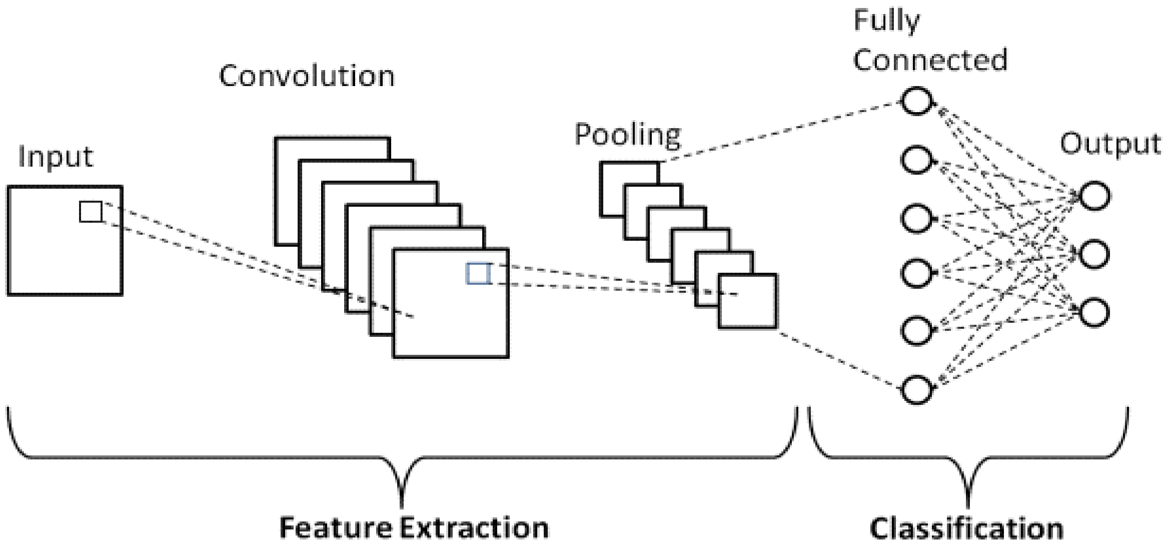


Figure 3.7 CNN Architecture

In Hindi news categorization, text is first transformed into numerical vectors, such as Word2Vec or BERT embeddings, before being fed into the CNN. The model processes these embeddings through convolutional layers to identify meaningful features and patterns that distinguish different categories. The final classification is done using fully connected layers that output probabilities for each category. CNNs are particularly useful for large datasets and can automatically learn hierarchical feature representations from text.

3.6 OVERVIEW OF TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (known as a corpus). It helps convert text data into numerical form, making it suitable for machine learning algorithms.
TF-IDF consists of two components:

Term Frequency (TF): Measures how frequently a word appears in a document.

Formula:

TF(t)=Number of times term t appears in a document/Total number of terms in the document

Inverse Document Frequency (IDF): Measures how unique or rare a word is across all documents in the corpus.

Formula:

IDF(t)=log(Total number of documents/Number of documents containing the term t+1)

The "+1" avoids division by zero if the term does not appear in any document.

The final **TF-IDF score** for a term ttt is the product of TF and IDF:

TF-IDF(t)=TF(t)×IDF(t)
TF-IDF Used in Hindi News Classification in following ways:

1. Text Preprocessing**:**
   - The Hindi news articles are preprocessed to remove noise. This involves tokenization, stopword removal, and normalization (e.g., converting text to lowercase).
   - Since TF-IDF works with tokenized text, Hindi text needs proper tokenization, respecting language-specific constructs.
2. Vector Representation:
   - Each word in a document is assigned a TF-IDF score.
   - The document is represented as a sparse vector, where each dimension corresponds to a unique term in the corpus.
3. Feature Extraction:
   - The TF-IDF matrix is generated, where rows represent documents, and columns represent terms with their TF-IDF scores.
   - This matrix is used as input features for machine learning models like Support Vector Machines (SVM), Logistic Regression, or even deep learning models.
4. Classification:
   - Machine learning algorithms use the TF-IDF features to learn patterns and classify the news articles into predefined categories like sports, politics, entertainment, or technology.`

Advantages of Using TF-IDF:

1. Relevance-Focused:
   - Words that are common across documents (e.g., "और" or "है" in Hindi) get lower scores, while unique and meaningful words get higher scores.
2. Simplicity:
   - Easy to compute and integrate into traditional machine learning pipelines.
3. Effectiveness in Sparse Data:
   - Performs well when working with high-dimensional, sparse text data, common in natural language tasks.

Limitations of TF-IDF:

1. Context Ignorance:
   - TF-IDF does not capture the semantic relationships between words or their contextual meanings (e.g., synonyms or polysemy).
2. Dimensionality Issues:
   - For large corpora, the number of unique terms can result in very high-dimensional feature spaces.
3. Static Scoring:
   - Scores are based solely on frequency and do not consider word order or grammar.

Extra Information: Extensions and Enhancements

1. Bigram and Trigram Models:
   - TF-IDF can be extended to consider sequences of words (e.g., "प्रधान मंत्री" in Hindi) by using bigrams or trigrams instead of single terms.
2. Domain-Specific Stopword Lists:
   - Creating custom stopword lists for Hindi ensures common but irrelevant terms are excluded from the TF-IDF computation.
3. Integration with Advanced Models**:**
   - TF-IDF can be used as input for more complex models, like combining it with embeddings (Word2Vec or BERT) for hybrid systems.
4. Low-Resource Language Adaptation:
   - TF-IDF is particularly useful for low-resource languages like Hindi, where large pre-trained embeddings may not be readily available.

TF-IDF is a foundational technique for converting text into numerical form for machine learning tasks. In Hindi news classification, it effectively highlights key terms in articles, enabling models to focus on meaningful words. While it has limitations in capturing deep contextual information, its simplicity and efficiency make it a powerful tool, especially when combined with machine learning algorithms for structured classification tasks.

3.7 SUMMARY

Machine Learning and Natural Language Processing are important subfields of Artificial Intelligence that have gained prominence in recent times. Combined with machine learning algorithm, NLP generate system that learn to perform task on their own and get better through experience. Natural Language Processing (NLP) and

Machine Learning (ML) are synergistically used in text classification to transform raw text into structured and actionable insights.

Text classification systems have evolved over decades, driven by the need to manage and analyze large volumes of textual data efficiently. From simple rule-based systems to modern deep learning techniques, the field has undergone significant transformations.

CHAPTER 4
METHODS AND MATERIALS

4.1 INTRODUCTION

The chapter above discusses various machine learning and deep learning models and algorithm. In this section, we discuss the methods implemented to carry out the various activities in the dataset cleaning and preprocessing stage and analyze the machine learning model for classifying Hindi newswire articles. This chapter is organized into Section 4.2, Section 4.3, and Section 4.4. Section 4.2 deals with pre-processing phase and Section 4.3 discuss linguistic tools used to make the Chhattisgarhi Hindi machine translator; in Section 4.4 conclusion is made.

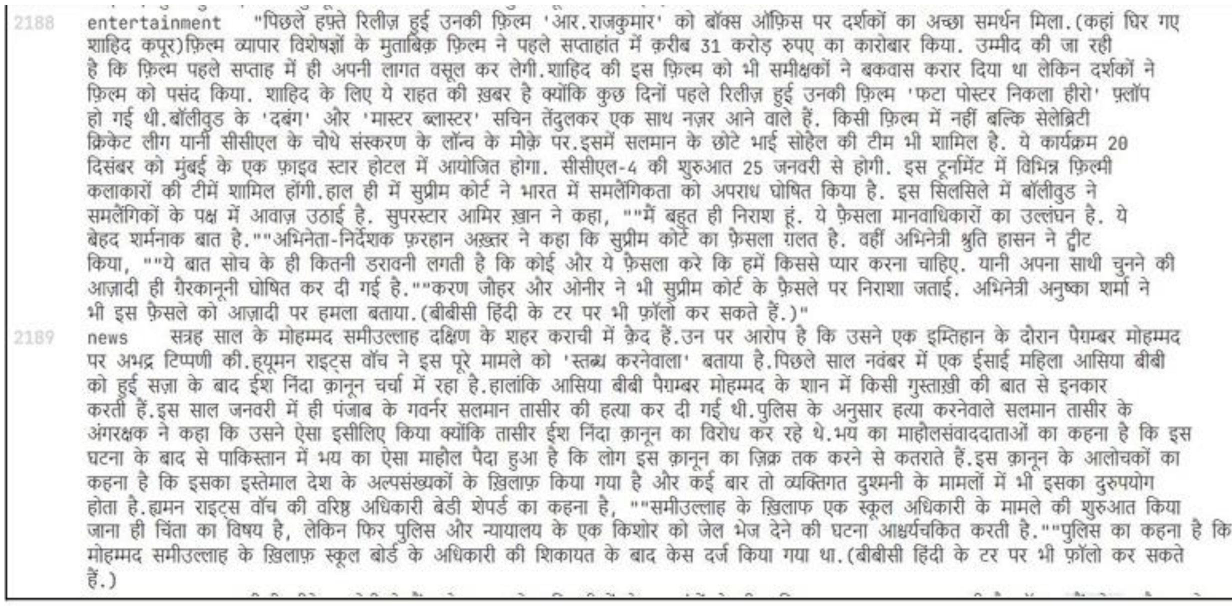
4.2 DATASET CLEANING AND PREPARATION

One of the most essential elements of NLP and machine learning is data. The System requires News Articles from Various Categories to Implement the Classification Model. Thus, these data were gathered from GitHub and Kaggle's BBC Hindi News Dataset. Some of the data were also extracted from the IIT Patna Disaster dataset. This dataset is typically utilized for more than just training. A single training data set that has previously been processed is usually divided into many segments to assess how well the model was trained. Typically, a testing data set is kept apart from the data for this particular reason. The overall dataset is about 3000 articles, and approximately 20 MB of data was used to train and test the models.

Text Pre-processing and feature extraction:

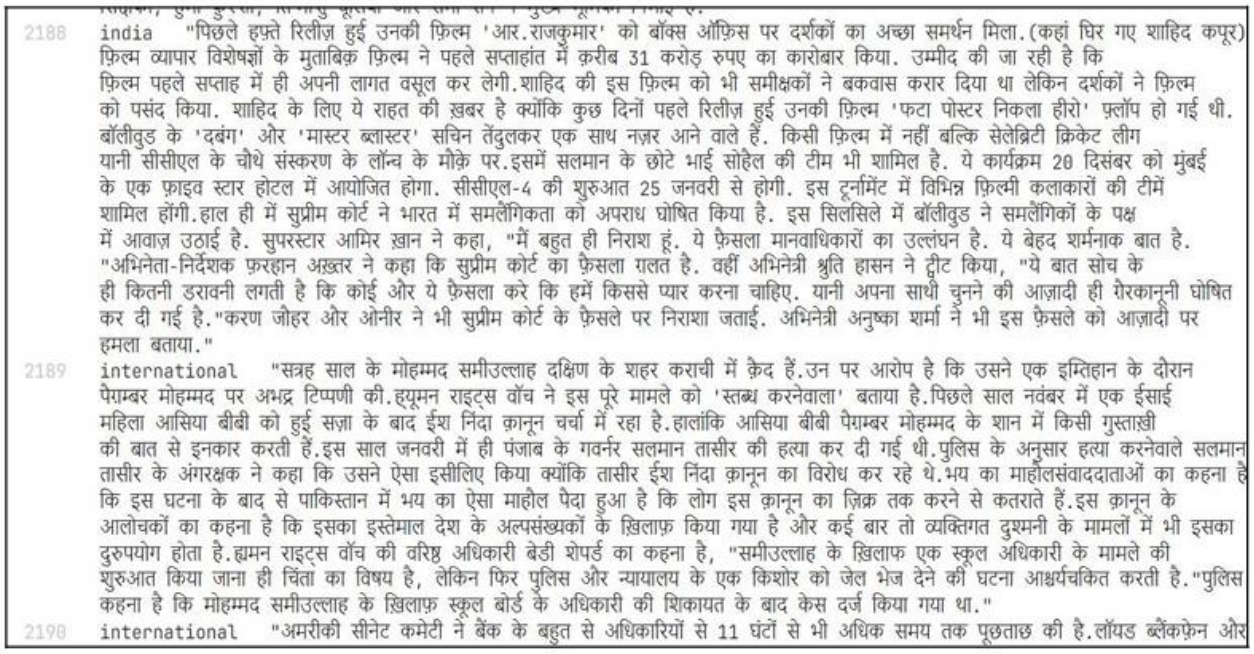
Pre-processing raw data is a crucial step in data preparation, improving the accuracy and efficacy of a machine learning model. To get the dataset ready for the classification model to be applied, the following procedures were taken:

i) Dataset Cleaning: Cleaning a dataset is the process of eliminating extraneous and noisy data. The main objective is to create a uniform format for the dataset. The data was cleaned by following the steps: Cleaning the dataset started with deleting undesired, irrelevant, and empty ad lines. There are no longer any blank lines in the article; the advertisement lines have been removed, and multi-row articles have been converted into one row. The next step is to enclose each article within double quotes.



```
entertainment   "पिछले हफ़्ते रिलीज़ हुई उनकी फ़िल्म 'आर.राजकुमार' को बॉक्स ऑफ़िस पर दर्शकों का अच्छा समर्थन मिला.(कहां घिर गए
शाहिद कपूर)फ़िल्म व्यापार विशेषज्ञों के मुताबिक़ फ़िल्म ने पहले सप्ताहांत में क़रीब 31 करोड़ रुपए का कारोबार किया. उम्मीद की जा रही
है कि फ़िल्म पहले सप्ताह में ही अपनी लागत वसूल कर लेगी.शाहिद की इस फ़िल्म को भी समीक्षकों ने बकवास करार दिया था लेकिन दर्शकों ने
फ़िल्म को पसंद किया. शाहिद के लिए ये राहत की ख़बर है क्योंकि कुछ दिनों पहले रिलीज़ हुई उनकी फ़िल्म 'फटा पोस्टर निकला हीरो' फ़्लॉप
हो गई थी.बॉलीवुड के 'दबंग' और 'मास्टर ब्लास्टर' सचिन तेंदुलकर एक साथ नज़र आने वाले हैं. किसी फ़िल्म में नहीं बल्कि सेलेब्रिटी
क्रिकेट लीग यानी सीसीएल के चौथे संस्करण के लॉन्च के मौक़े पर.इसमें सलमान के छोटे भाई सोहैल की टीम भी शामिल है. ये कार्यक्रम 20
दिसंबर को मुंबई के एक फ़ाइव स्टार होटल में आयोजित होगा. सीसीएल-4 की शुरुआत 25 जनवरी से होगी. इस टूर्नामेंट में विभिन्न फ़िल्मी
कलाकारों की टीमें शामिल होंगी.हाल ही में सुप्रीम कोर्ट ने भारत में समलैंगिकता को अपराध घोषित किया है. इस सिलसिले में बॉलीवुड ने
समलैंगिकों के पक्ष में आवाज़ उठाई है. सुपरस्टार आमिर ख़ान ने कहा, ""मैं बहुत ही निराश हूं. ये फ़ैसला मानवाधिकारों का उल्लंघन है. ये
बेहद शर्मनाक बात है.""अभिनेता-निर्देशक फ़रहान अख़्तर ने कहा कि सुप्रीम कोर्ट का फ़ैसला ग़लत है. वहीं अभिनेत्री श्रुति हासन ने ट्वीट
किया, ""ये बात सोच के ही कितनी डरावनी लगती है कि कोई और ये फ़ैसला करे कि हमें किससे प्यार करना चाहिए. यानी अपना साथी चुनने की
आज़ादी ही ग़ैरकानूनी घोषित कर दी गई है.""करण जौहर और ओनीर ने भी सुप्रीम कोर्ट के फ़ैसले पर निराशा जताई. अभिनेत्री अनुष्का शर्मा ने
भी इस फ़ैसले को आज़ादी पर हमला बताया.(बीबीसी हिंदी के टर पर भी फ़ॉलो कर सकते हैं.)"
news    सत्रह साल के मोहम्मद समीउल्लाह दक्षिण के शहर कराची में क़ैद हैं.उन पर आरोप है कि उसने एक इम्तिहान के दौरान पैग़म्बर मोहम्मद
पर अभद्र टिप्पणी की.ह्यूमन राइट्स वॉच ने इस पूरे मामले को 'स्तब्ध करनेवाला' बताया है.पिछले साल नवंबर में एक ईसाई महिला आसिया बीबी
को हुई सज़ा के बाद ईश निंदा क़ानून चर्चा में रहा है.हालांकि आसिया बीबी पैग़म्बर मोहम्मद के शान में किसी गुस्ताख़ी की बात से इनकार
करती हैं.इस साल जनवरी में ही पंजाब के गवर्नर सलमान तासीर की हत्या कर दी गई थी.पुलिस के अनुसार हत्या करनेवाले सलमान तासीर के
अंगरक्षक ने कहा कि उसने ऐसा इसीलिए किया क्योंकि तासीर ईश निंदा क़ानून का विरोध कर रहे थे.भय का माहौलसंवाददाताओं का कहना है कि इस
घटना के बाद से पाकिस्तान में भय का ऐसा माहौल पैदा हुआ है कि लोग इस क़ानून का ज़िक्र तक करने से कतराते हैं.इस क़ानून के आलोचकों का
कहना है कि इसका इस्तेमाल देश के अल्पसंख्यकों के ख़िलाफ़ किया गया है और कई बार तो व्यक्तिगत दुश्मनी के मामलों में भी इसका दुरुपयोग
होता है.ह्यमन राइट्स वॉच की वरिष्ठ अधिकारी बेडी शेपर्ड का कहना है, ""समीउल्लाह के ख़िलाफ एक स्कूल अधिकारी के मामले की शुरुआत किया
जाना ही चिंता का विषय है, लेकिन फिर पुलिस और न्यायालय के एक किशोर को जेल भेज देने की घटना आश्चर्यचकित करती है.""पुलिस का कहना है कि
मोहम्मद समीउल्लाह के ख़िलाफ़ स्कूल बोर्ड के अधिकारी की शिकायत के बाद केस दर्ज किया गया था.(बीबीसी हिंदी के टर पर भी फ़ॉलो कर सकते
हैं.)
```

Figure 4.1: Dataset Before removing blank spaces and unwanted lines

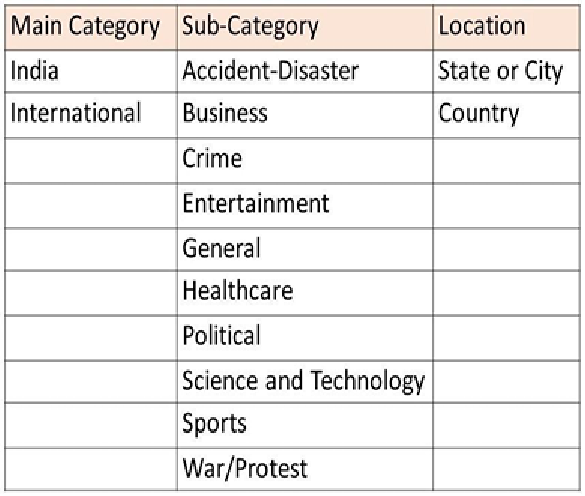


```
india   "पिछले हफ़्ते रिलीज़ हुई उनकी फ़िल्म 'आर.राजकुमार' को बॉक्स ऑफ़िस पर दर्शकों का अच्छा समर्थन मिला.(कहां घिर गए शाहिद कपूर)
फ़िल्म व्यापार विशेषज्ञों के मुताबिक़ फ़िल्म ने पहले सप्ताहांत में क़रीब 31 करोड़ रुपए का कारोबार किया. उम्मीद की जा रही है कि
फ़िल्म पहले सप्ताह में ही अपनी लागत वसूल कर लेगी.शाहिद की इस फ़िल्म को भी समीक्षकों ने बकवास करार दिया था लेकिन दर्शकों ने फ़िल्म
को पसंद किया. शाहिद के लिए ये राहत की ख़बर है क्योंकि कुछ दिनों पहले रिलीज़ हुई उनकी फ़िल्म 'फटा पोस्टर निकला हीरो' फ़्लॉप हो गई थी.
बॉलीवुड के 'दबंग' और 'मास्टर ब्लास्टर' सचिन तेंदुलकर एक साथ नज़र आने वाले हैं. किसी फ़िल्म में नहीं बल्कि सेलेब्रिटी क्रिकेट लीग
यानी सीसीएल के चौथे संस्करण के लॉन्च के मौक़े पर.इसमें सलमान के छोटे भाई सोहैल की टीम भी शामिल है. ये कार्यक्रम 20 दिसंबर को मुंबई
के एक फ़ाइव स्टार होटल में आयोजित होगा. सीसीएल-4 की शुरुआत 25 जनवरी से होगी. इस टूर्नामेंट में विभिन्न फ़िल्मी कलाकारों की टीमें
शामिल होंगी.हाल ही में सुप्रीम कोर्ट ने भारत में समलैंगिकता को अपराध घोषित किया है. इस सिलसिले में बॉलीवुड ने समलैंगिकों के पक्ष
में आवाज़ उठाई है. सुपरस्टार आमिर ख़ान ने कहा, "मैं बहुत ही निराश हूं. ये फ़ैसला मानवाधिकारों का उल्लंघन है. ये बेहद शर्मनाक बात है.
"अभिनेता-निर्देशक फ़रहान अख़्तर ने कहा कि सुप्रीम कोर्ट का फ़ैसला ग़लत है. वहीं अभिनेत्री श्रुति हासन ने ट्वीट किया, "ये बात सोच के
ही कितनी डरावनी लगती है कि कोई और ये फ़ैसला करे कि हमें किससे प्यार करना चाहिए. यानी अपना साथी चुनने की आज़ादी ही ग़ैरकानूनी घोषित
कर दी गई है."करण जौहर और ओनीर ने भी सुप्रीम कोर्ट के फ़ैसले पर निराशा जताई. अभिनेत्री अनुष्का शर्मा ने भी इस फ़ैसले को आज़ादी पर
हमला बताया."
international   "सत्रह साल के मोहम्मद समीउल्लाह दक्षिण के शहर कराची में क़ैद हैं.उन पर आरोप है कि उसने एक इम्तिहान के दौरान
पैग़म्बर मोहम्मद पर अभद्र टिप्पणी की.ह्यूमन राइट्स वॉच ने इस पूरे मामले को 'स्तब्ध करनेवाला' बताया है.पिछले साल नवंबर में एक ईसाई
महिला आसिया बीबी को हुई सज़ा के बाद ईश निंदा क़ानून चर्चा में रहा है.हालांकि आसिया बीबी पैग़म्बर मोहम्मद के शान में किसी गुस्ताख़ी
की बात से इनकार करती हैं.इस साल जनवरी में ही पंजाब के गवर्नर सलमान तासीर की हत्या कर दी गई थी.पुलिस के अनुसार हत्या करनेवाले सलमान
तासीर के अंगरक्षक ने कहा कि उसने ऐसा इसीलिए किया क्योंकि तासीर ईश निंदा क़ानून का विरोध कर रहे थे.भय का माहौलसंवाददाताओं का कहना है
कि इस घटना के बाद से पाकिस्तान में भय का ऐसा माहौल पैदा हुआ है कि लोग इस क़ानून का ज़िक्र तक करने से कतराते हैं.इस क़ानून के
आलोचकों का कहना है कि इसका इस्तेमाल देश के अल्पसंख्यकों के ख़िलाफ़ किया गया है और कई बार तो व्यक्तिगत दुश्मनी के मामलों में भी इसका
दुरुपयोग होता है.ह्यमन राइट्स वॉच की वरिष्ठ अधिकारी बेडी शेपर्ड का कहना है, "समीउल्लाह के ख़िलाफ एक स्कूल अधिकारी के मामले की
शुरुआत किया जाना ही चिंता का विषय है, लेकिन फिर पुलिस और न्यायालय के एक किशोर को जेल भेज देने की घटना आश्चर्यचकित करती है."पुलिस
कहना है कि मोहम्मद समीउल्लाह के ख़िलाफ़ स्कूल बोर्ड के अधिकारी की शिकायत के बाद केस दर्ज किया गया था."
international   "अमरीकी सीनेट कमेटी ने बैंक के बहुत से अधिकारियों से 11 घंटों से भी अधिक समय तक पूछताछ की है.लॉयड ब्लैंकफ़ेन और
```

Figure 4.2 Dataset After Removing Blank Spaces and Unwanted Lines

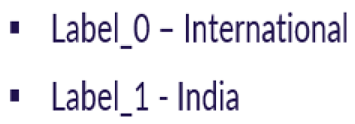
ii) Dataset Classification:

Table shows the categories and subcategories created in the dataset for the classification process. Figure shows the categories and subcategories converted to Hindi Language. Then, the articles are manually tagged in each data row according to the type of categories and sub-categories. Each article unit is categorized into three columns, as shown in figure.



Main Category	Sub-Category	Location
India	Accident-Disaster	State or City
International	Business	Country
	Crime	
	Entertainment	
	General	
	Healthcare	
	Political	
	Science and Technology	
	Sports	
	War/Protest	

Table 4.1: Summary of Categorization



- Label_0 – International
- Label_1 - India

Figure 4.3: Classification to tag news articles in Hindi

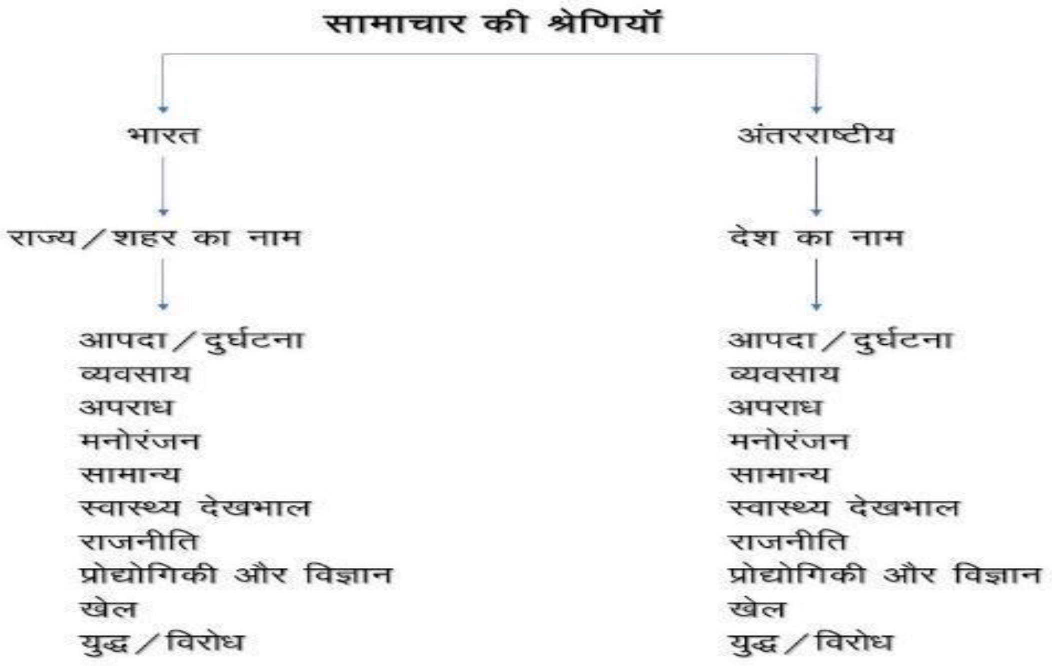


Figure 4.4 Classification to tag news articles in Hindi

iii) Preprocessing:

Machine Learning and NLP techniques can efficiently organize Hindi Newswire articles into various sections. Tokenization, Embedding, stopword removable, Text filtering & cleaning, and vectorization are techniques used in binary classification.

4.3 Hybrid Classifier Models

The key idea behind these hybrid models is to leverage the feature extraction capabilities of CNN and combine them with the classification strengths of both Logistic Regression (LR) and Support Vector Machine (SVM). CNN is used to automatically learn and extract meaningful features from the data, while LR and SVM are applied for the final classification based on these features. This hybrid approach integrates deep learning and traditional machine learning techniques to create a model that can capture complex patterns while maintaining robust classification performance. BERT is a state-of-the-art pre-trained transformer model designed to understand the context of words in a sentence by processing text bidirectionally. It provides deep contextual embeddings that capture the semantic and syntactic nuances of the text.

4.3.1 CNN-SVM-LR

FIRST LAYER

CNN Model: Train the CNN model using sequences and extract its features.

Input: Tokenizing and padding the sequences for CNN (Raw splited data).

CNN Features: The output of the dense layer is considered the extracted feature representation of the input text.

SECOND LAYER

Combining TF-IDF features with the CNN features

SVM Model (RBF Kernel): Train the SVM model using combined TF-IDF features and CNN features.

Input: Combined features of TF-IDF and CNN.

SVM features: Features Extracted using predict_proba.

Logistic Regression Model:Train the LR model using combined TF-IDF& CNN features.

Input: Combined features of TF-IDF and CNN.

LR features: Features Extracted using predict_proba

THIRD(FINAL) LAYER

Then Combine SVM features with LR features.

Final Model- LR model: Final LR model is trained by utilizing the combined features of SVM and LR.

```
OUTPUT:

Train Precision: 00.9899

Test Accuracy: 00.6111

Precision: 00.6368
```

```
Recall: 00.6111

F1-score: 00.6128

Training Loss: [8.239630699157715, 4.46433687210083, 4.418089866638184, 10.207816123962402,
8.60403823852539, 4.208271503448486, 4.13950777053833, 4.11625337600708, 4.108708381652832,
4.106342315673828]

Validation Loss: [7.49569845199585, 4.605170726776123, 4.598260879516602, 10.181602478027344,
8.371447563171387, 5.844483375549316, 5.71553373336792, 5.710078716278076,
5.6995463371276855, 5.687788486480713]
```

4.3.2 SVM-CNN-LR

FIRST LAYER

SVM Model (RBF Kernel): Train the SVM model using TF-IDF features and extract predict_proba values as features.

Input: TF-IDF Vectorized data.

SVM features: Features Extracted using predict_proba function.

CNN Model: Train the CNN model using sequences and extract its features.

Input: Tokenizing and padding the sequences for CNN.

CNN Features: The output of the dense layer (or the global max-pooling layer) is considered the extracted feature representation of the input text.

SECOND(FINAL) LAYER

Then Combine SVM features with CNN features.

Logistic Regression Model: Combine the features from the SVM and CNN models to train the final LR model.

```
OUTPUT:

Test Precision: 00.6066

Train Accuracy: 00.9884

F1-score: 00.6007

Precision: 00.6240

Recall: 00.6066

Training Loss: [8.46601390838623, 4.955399036407471, 4.431170463562012, 4.490259170532227,
4.865240097045898, 4.615891456604004, 4.605289459228516, 4.599568843841553,
4.596526145935059, 4.596526622772217]

Validation Loss: [8.006915092468262, 4.614402770996094, 4.605170726776123, 4.605170726776123,
4.611858367919922, 4.605170726776123, 4.605170726776123, 4.605170726776123,
4.605170726776123, 4.605170726776123]
```

4.3.3 LR-SVM-CNN

FIRST LAYER

LR Model (RBF Kernel): Train the SVM model using TF-IDF features and extract predict_proba values as features.

- Input: TF-IDF Vectorized data.
- LR features: Features Extracted using predict_proba function.

SECOND LAYER

RBF Kernel (SVM Model): The SVM model is trained using LR features, also extract predict_proba values as features.

Input: LR Features.

LR features: Features Extracted using predict_proba function.

Then Combine LR features with SVM features.

CNN Model: Train the CNN model using sequences and extract its features.

Input: Tokenizing and padding the sequences for CNN (Raw splited data)

CNN Features: The result of the dense layer is considered the extracted feature representation of the input text.

Then Combine LR-SVM combined features with CNN features.

THIRD LAYER (Final Layer)

Logistic Regression Model: Combine the features from the LR-SVM and CNN models to train the final Logistic Regression model. CNN features combined with the LR-SVM features correctly for final classification.

```
OUTPUT:

Test Precision: 00.5195

Train Accuracy: 00.9985

Recall: 00.5195

Precision: 00.5410

F1-score: 00.4988

Training Loss: [2.152841091156006, 2.008726119995117, 1.723357915878296, 1.2623515129089355,
0.7551372051239014, 0.34005656838417053, 0.1331361085176468, 0.060049522668123245,
0.03201261907815933, 0.027389468625187874]
```

```
Validation Loss: [2.109464168548584, 2.012880802154541, 1.797243356704712, 1.6646088361740112,
1.6226277351379395,      1.6894822120666504,      1.7818773984909058,      1.8656383752822876,
1.8496516942977905, 1.9149985313415527]
```

4.3.4 Bert-LR-SVM

First Layer: BERT for Feature Extraction

BERT Encoding and Tokenization: The text data is encoded into input IDs and attention masks using BERT's tokenizer. To extract features, these are then loaded into the BERT. Training BERT: BERT model is trained as a classifier for the given text classification task. It outputs features for each text input. Extracting BERT Features: After training, BERT's output features are extracted for both training and test datasets. These features represent the text inputs in a high-dimensional space, capturing contextual information.

Second Layer: Logistic Regression

TF-IDF Vectorization: In parallel, the text data is vectorized using TF-IDF to capture inverse document frequency and term frequency. Features Combination: The TF-IDF features and BERT features are combined into a single feature set for each text input. Training LR: A LR model is trained by utilizing the combined feature set. The model learns to predict class probabilities for each text input.

Third Layer: SVM

Training SVM with Logistic Regression Outputs: The probabilities output by the LR model are used as inputs to train an SVM model. The SVM model learns to classify the texts based on these probabilities. Final Predictions: The SVM model outputs probabilities, which are then used by a final Logistic Regression model to make the final class predictions.

```
Output:

Train Precision: 00.6745

Test Accuracy: 00.5676

Precision: 00.5564

F1-score: 00.5336

Recall: 00.5676
```

```
Training Loss: [4.013143062591553, 2.5883641242980957, 2.6182050704956055, 2.550568103790283,
2.5388870239257812]

Validation Loss: [2.610121726989746, 2.766256332397461, 2.618936061859131, 2.594259262084961,
2.596334934234619]
```

4.3.5 Bert-SVM-LR

First Layer: BERT Feature Extraction

BERT Tokenization and Encoding: The BERT tokenizer is applied to tokenize the input text. The text are encoded into input IDs and attention masks using the encode_with_bert_batch function.

BERT Model for Classification: A pre-trained BERT model (TF Bert For Sequence Classification) is loaded and compiled. The model is trained using the training data to perform text classification. After training, BERT features are extracted from the model for each input text using the extract_bert_features function

Second Layer: SVM Feature Extraction

Feature Combination: TF-IDF features and BERT features are combined for each input text. The combined features serve as input to the SVM model.

SVM Model Training: An SVM model (SVC) is trained on the combined features. The SVM model predicts probabilities for each class on the training and test data.

Third Layer: LR

LR Model Training: A LR model is also trained on the same combined features used by the SVM. It predicts probabilities for each class on the training and test data. Combining SVM and LR.

Final Model Training: The final LR model is trained on the combined SVM and Logistic Regression probabilities.

Output:

```
Train Precision: 00.9940

Test Accuracy: 00.5916

F1-score: 00.5839

Precision: 00.6220

Recall: 00.5916
```

```
Training Loss: [4.613874912261963, 4.1143364906311035, 3.1798369884490967, 2.588371515274048,
2.2668204307556152, 2.196584701538086, 2.1609277725219727, 2.1313040256500244,
2.1122031211853027, 2.1364340782165527]

Validation Loss: [4.159295558929443, 4.166423320770264, 2.3653995990753174, 2.178743839263916,
2.3532354831695557, 2.200831651687622, 2.1119728088378906, 2.1243607997894287,
2.117915153503418, 2.1290881633758545]
```

4.3.6 LR-SVM-Bert

First Layer: Logistic Regression Feature Extraction

TF-IDF features are created from the input text data. To categorize the text, a Logistic Regression model is trained by employing these TF-IDF attributes. The probabilities predicted by this LR model are used as features for the next layer.

Second Layer: SVM Feature Extraction

The LR features (from the first layer) are combined with the original TF-IDF features. A Support Vector Machine (SVM) model have been trained on this combined feature set. The decision function values (output) from the SVM model are used as features for the next layer.

Third Layer: BERT Feature Extraction and Final Classification

The BERT model tokenizes and encodes the text data to extract contextual features. These BERT features are combined with the features from the second layer (LR + SVM). Trained the final Logistic Regression model on this comprehensive feature set to perform the final classification.

```
Output:

 Train Accuracy: 0.9884          Test Accuracy: 0.6096

 Train Loss: 0.1011               Test Loss: 1.8284

 Train F1 Score: 0.9884           Test F1 Score: 0.6065

 Train Recall: 0.9884             Test Recall: 0.6096

 Train Precision: 0.9884         Test Precision: 0.6185
```

4.4 HLM-CLS: HYBRID LEARNING MODEL USING CNN-LR-SVM

Combining CNN-LR-SVM into a single pipeline for text categorization leverages the strengths of both deep learning and classical machine learning models. The primary aim of this hybrid approach is to handle complex decision boundaries, effectively categorize text, and capture local features. It is particularly suited for applications that require both detailed feature extraction and sophisticated classification. This model proves especially useful when deep learning techniques alone are insufficient for classification refinement or when traditional models struggle to capture intricate patterns.

4.4.1 Algorithm

Combining CNN features with TF-IDF provides a comprehensive representation of text by capturing both shallow statistical relationships (TF-IDF) and deep semantic patterns (CNN), enhancing the model's ability to understand and classify the data effectively. TF-IDF captures word frequency-based features, emphasizing important but uncommon terms across documents. It provides a sparse, statistical bag-of-words representation, useful for identifying key terms but lacking word order or semantic meaning. CNNs extract higher-level patterns from sequences by capturing local dependencies and spatial hierarchies. Unlike TF-IDF, they identify word combinations, semantic, and syntactic features, revealing contextual meaning in text.

```
Input Processing:

    1. Import Suitable Libraries:
        - Import suitable and relevant libraries.
    2. Load Datasets:
        - Load datasets representing different news categories into pandas DataFrames.
        - Assign labels to each category of news.
    3. Combine Datasets:
        - Concatenate all the category DataFrames into a single DataFrame for further
          processing.
    4. Define Features and Labels:
        - Separate the news articles `$\boldsymbol{X}$` and their corresponding labels `$\boldsymbol{y}$`.
                  $\boldsymbol{X = news['news\_articles']}$
                  $\boldsymbol{y = news['label']}$
    5. Define Custom Tokenizer and Stop Words:
        - Define a list of custom stop words.
        - Create a custom tokenizer that splits text into tokens based on spaces.
Feature Extraction:
```

6. Extract importance of each word using TF-IDF Vectorization:
    - Initialize the `$\boldsymbol{TfidfVectorizer}$` with custom stop words and tokenizer.
    - Divide the dataset into two sets i.e. training and testing sets.
    - Transform the training and testing text data into TF-IDF features.

<u>TF-IDF Calculation:</u>

For a given term `$\boldsymbol{t}$`, document `$\boldsymbol{d}$`, and document corpus `$\boldsymbol{D}$`:

Term Frequency (TF):

$$\boldsymbol{TF(t,d) = (Number\ of\ times\ term\ t\ appears\ in\ document\ d) / (Total\ number\ of\ terms\ in\ document\ d)}$$

Inverse Document Frequency (IDF):

$$\boldsymbol{IDF(t,D) = log(N / |\{d \in D : t \in d\}|)}$$

where `$\boldsymbol{N}$` represents the total number of documents within the corpus.

TF-IDF:

$$\boldsymbol{TF-IDF(t,d,D) = TF(t,d) \times IDF(t,D)}$$

7. Tokenization and Padding for CNN:
    - Convert the text data into sequences of integers through tokenization.
    - Pad the sequences to achieve a consistent length across all input data.

8. Build the CNN Model:
    - Design a Convolutional Neural Network (CNN) incorporating an embedding component, a convolutional component, and a global max-pooling component.

    - Compile and train the CNN model on the training data.

Convolution Operation:

Given an input sequence `
$\boldsymbol{X}$` of length `$\boldsymbol{n}$`, and a filter `$\boldsymbol{w}$` of size `$\boldsymbol{k}$`:

$$\boldsymbol{Convolution\ Output = X * w = \sum(i = 1\ to\ k)\ X[i] \times w[i]}$$

Global Max-Pooling:

For the convolution output `Z`, the global max-pooling operation computes:

$$\boldsymbol{Global\ Max-Pooling(Z) = max(Z)}$$

9. Extract relationship between words using CNN:
    - Use the trained CNN model to extract features from the training and testing datasets.

$$\boldsymbol{CNN\ Features = CNN(X\_train\_pad)}$$

<u>Feature Combination:</u>

10. Combine importance of words and relationship between word extracted using TF-IDF and CNN:
    - Horizontally stack the TF-IDF features with the CNN-extracted features to create combined feature sets for training and testing.

$$\boldsymbol{Train\ Features = TF-IDF\ Features \oplus CNN\ Features}$$
$$\boldsymbol{Test\ Features = TF-IDF\ Features \oplus CNN\ Features}$$

where $\oplus$ denotes concatenation

Feature Reduction:
11. Define SVM and Logistic Regression Models for feature reduction: - Initialize an SVM model and a Logistic Regression model.
12. Train SVM model to generate optimal hyperplane and Logistic Regression model to capture linear relationship: - Train both models on the combined features of TF-IDF and CNN. *SVM Decision Function:* The decision function for SVM is given by: $\boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{sign}(\boldsymbol{w} \times \boldsymbol{X} + \boldsymbol{b})$ where \`$w$\` is the weight vector, and \`$b$\` is the bias. *Logistic Regression Probability:* The probability prediction in Logistic Regression is given by the sigmoid function: $\boldsymbol{P}(\boldsymbol{y} = \boldsymbol{1} \| \boldsymbol{X}) = \boldsymbol{\sigma}(\boldsymbol{w} \times \boldsymbol{X} + \boldsymbol{b}) = \boldsymbol{1} / (\boldsymbol{1} + \boldsymbol{e}\hat{} - (\boldsymbol{w} \times \boldsymbol{X} + \boldsymbol{b}))$
13. Predict Probabilities: - Use the trained SVM and Logistic Regression models to predict probabilities for both training and testing datasets.
14. Combine SVM and Logistic Regression Probabilities to get reduced feature set: - Horizontally stack the probabilities from the SVM and Logistic Regression models to reduce dimensionality and get a final feature set. $\boldsymbol{Y\_combined} = \boldsymbol{SVM\ Probabilities} \oplus \boldsymbol{Logistic\ Regression\ Probabilities}$
Final Model and Evaluation:
15. Train a Final Base Model on reduced final feature set: - Fit a final Support Vector Machine (SVM) model on the combined probabilities.
16. Make Predictions: - Utilize the final Support Vector Machine (SVM) model to generate predictions for both the training and test data. $\hat{y} = \boldsymbol{f}(\boldsymbol{Y\_combined})$

17.	Calculate Metrics: - Compute the following metrics: *Accuracy:* $\boldsymbol{Accuracy = (Number\ of\ correct\ predictions) / (Total\ number\ of\ predictions)}$ *Precision:* $\boldsymbol{Precision = (True\ Positives) / (True\ Positives + False\ Positives)}$ *Recall:* $\boldsymbol{Recall = (True\ Positives) / (True\ Positives + False\ Negatives)}$ *F1-Score:* $\boldsymbol{F1 - Score = 2 \times (Precision \times Recall) / (Precision + Recall)}$ - Compute the training and validation loss history from the CNN model training.
18.	Output Results: - Print the calculated metrics and loss values to evaluate model performance.

4.4.2 Flow diagram

The CNN-LR-SVM model shown in figure 2 represents an advanced strategy that integrates both machine learning and deep learning for text categorization, utilizing feature extraction through CNN, followed by classification with LR and SVM, often incorporating TF-IDF vectorization for additional feature representation.

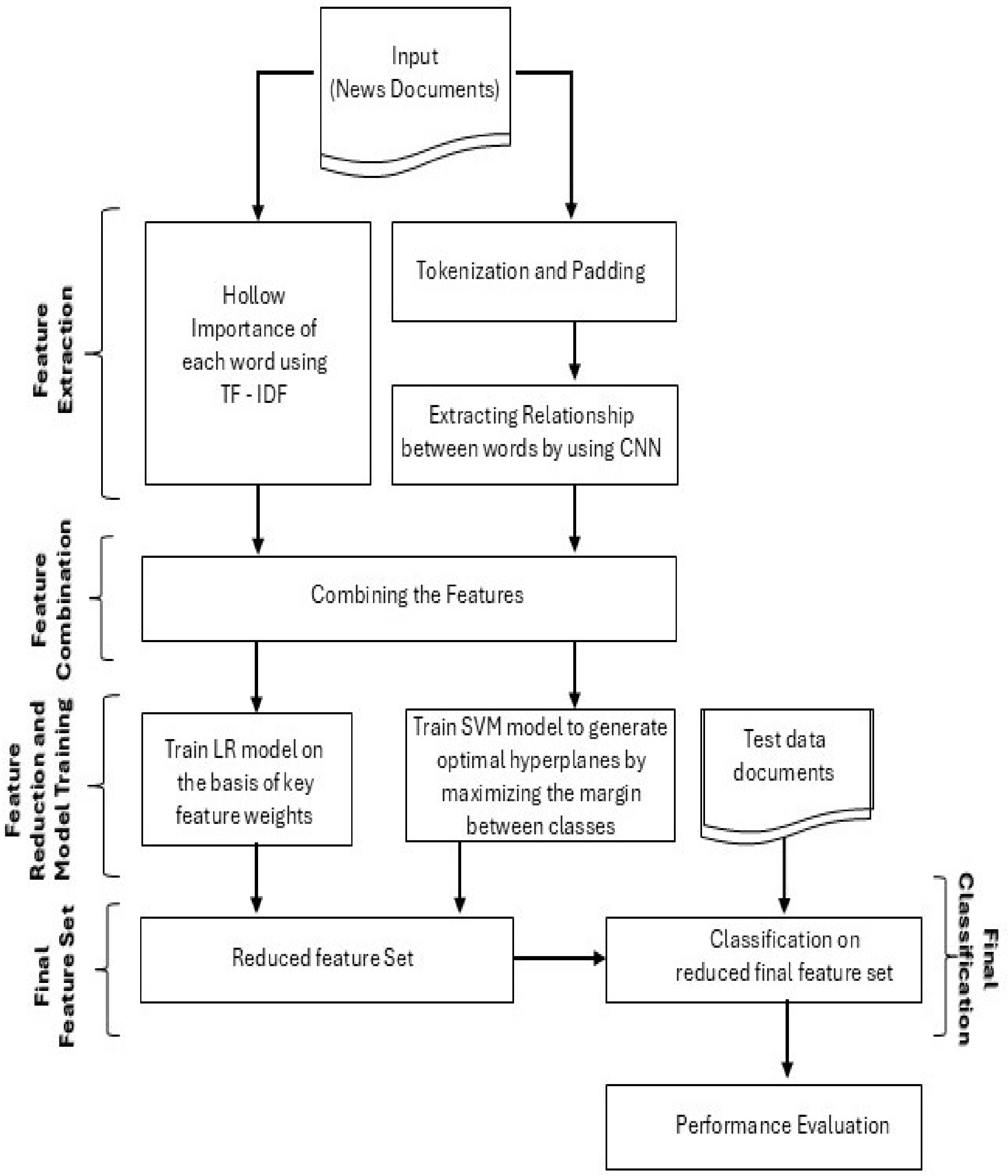

Input (News Documents)
Feature Extraction
Hollow Importance of each word using TF - IDF
Tokenization and Padding
Extracting Relationship between words by using CNN
Feature Combination
Combining the Features
Feature Reduction and Model Training
Train LR model on the basis of key feature weights
Train SVM model to generate optimal hyperplanes by maximizing the margin between classes
Test data documents
Final Feature Set
Reduced feature Set
Classification on reduced final feature set
Final Classification
Performance Evaluation

Figure 4.5 Flow diagram of CLS

4.5 HLM-SBL: HYBRID LEARNING MODEL USING SVM-Bert-LR

Combining SVM-BERT-LR models into a single pipeline for text categorization leverages the strengths of both deep learning and classical machine learning models. The SVM-BERT-LR model is a useful hybrid technique for text categorization that combines advanced deep learning research with dependable conventional machine learning techniques. Utilizing the contextual embeddings of BERT, the classification efficiency of LR, and the decision-refinement power of SVM, this model provides a comprehensive solution for complex text categorization tasks. A text categorization model that combines SVM, BERT, and Logistic Regression (LR) is intended to maximize the advantages of each element. This hybrid model makes use of LR's linear classification skills, SVM's ability to handle complex decision boundaries, and BERT's contextual embeddings.

4.5.1 Algorithm

```
Input Processing:

1. Import Suitable Libraries:
        - Import suitable and relevant libraries.

2. Load Datasets:
        - Load datasets representing different news categories into pandas DataFrames.
        - Assign labels to each category of news.

3. Combine Datasets:
        - Concatenate all the category DataFrames into a single DataFrame for further
          processing.

4. Define Features and Labels:
        - Separate the news articles `X` and their corresponding labels `y`.
                    $X = news['news\_articles']$
                    $y = news['label']$

5. Define Custom Tokenizer and Stop Words:
        - Define a list of custom stop words.
        - Create a custom tokenizer that splits text into tokens based on spaces.

Feature Extraction:
```

6. **Identify basic word patterns and dependencies using TF-IDF Vectorization:**
    - **Initialize the `TfidfVectorizer` with custom stop words and tokenizer.**
    - **TF-IDF helps convert textual data into vectors.**
    - **Divide the dataset into two sets i.e. training and testing sets.**
    - **Transform the training and testing text data into TF-IDF features.**

<u>TF-IDF Calculation:</u>
For a given term `t`, document `d`, and document corpus `D`:
Term Frequency (TF):

$$\mathbf{TF(t, d) = (Frequenty\ of\ term\ t\ appears\ in\ document\ d) / (Total\ number\ of\ terms\ in\ document\ d)}$$

Inverse Document Frequency (IDF):

$$\mathbf{IDF(t, D) = log(N / |\{d \in D : t \in d\}|)}$$

where `N` represents the total number of documents within the corpus.
TF-IDF:

$$\mathbf{TF - IDF(t, d, D) = TF(t, d) \times IDF(t, D)}$$

7. **SVM for Feature Extraction:**
    - **Identify Hyperplane by applying margin-based classification.**
    - **Initialize an SVM model with RBF kernel:**

$$\mathbf{K(v_i, v_j) = exp(-\gamma|v_i - v_j|^2)}$$

    - **Parameters used:**

$$\boldsymbol{\gamma = 0.001, \quad C = 1000}$$

8. **BERT for Feature Extraction:**
    - **Load the pre-trained BERT model and its tokenizers (bert-base-uncased).**
    - **Encode the textual data using the BERT tokenizer to convert them into token IDs and attention masks:**

$$\mathbf{X_{train\_ids}, X_{train\_masks} = encode\_with\_bert\_batch(X_{train}.tolist(), tokenizer, max\_length = 100, batch\_size = 32)}$$

    - **Using the BERT model to extract features from the token IDs:**

$$\mathbf{F_{train\_bert} = extract\_bert\_features X_{train\_ids}, X_{train\_masks}. bert\_model\_for\_classification)}$$

    - **Extract Deep Embedding and capture semantic relationship between words in a sentence using BERT.**
    - **BERT embeddings encode deep semantic information, helping in resolves ambiguities from TF-IDF stage.**

<u>Feature Combination:</u>

9. **Combine importance of words and relationship between word extracted using SVM and BERT:**
    - **Horizontally stack the SVM features with the BERT extracted features to create combined feature sets for training and testing.**

$$\mathbf{F_{train\_final} = np.hstack\left(\left(F_{train\_combined}, F_{train\_bert}\right)\right)}$$
$$\mathbf{F_{test\_final} = np.hstack\left(\left(F_{test\_combined}, F_{test\_bert}\right)\right)}$$

Final Feature Set:

10. Define Logistic Regression Models and BERT's deep embedding for final feature set:
- Initialize an Logistic Regression model and a deep embedding from BERT model.

11. From BERT model extract deep embedding and Logistic Regression model to capture linear relationship:
- Logistic Regression performs final classification on embedding extracted by BERT model.
- Initialize a Logistic Regression model for cross-entropy loss.

$$\textbf{Cross-Entropy-Loss} = -\frac{1}{N}\sum_{x=1}^{N}\sum_{r=1}^{Z} y_{x,r} \log\left(\widehat{p_{x,r}}\right)$$

- Here, Z = Total number of classes, $y_{x,r}$ = {0, 1}, r = correct classification for observation x and $p_{x,r}$ = predicted probability

12. Train Logistic Regression model:
- Train Logistic Regression model on the entire combined training features.

Final Classification and Evaluation:

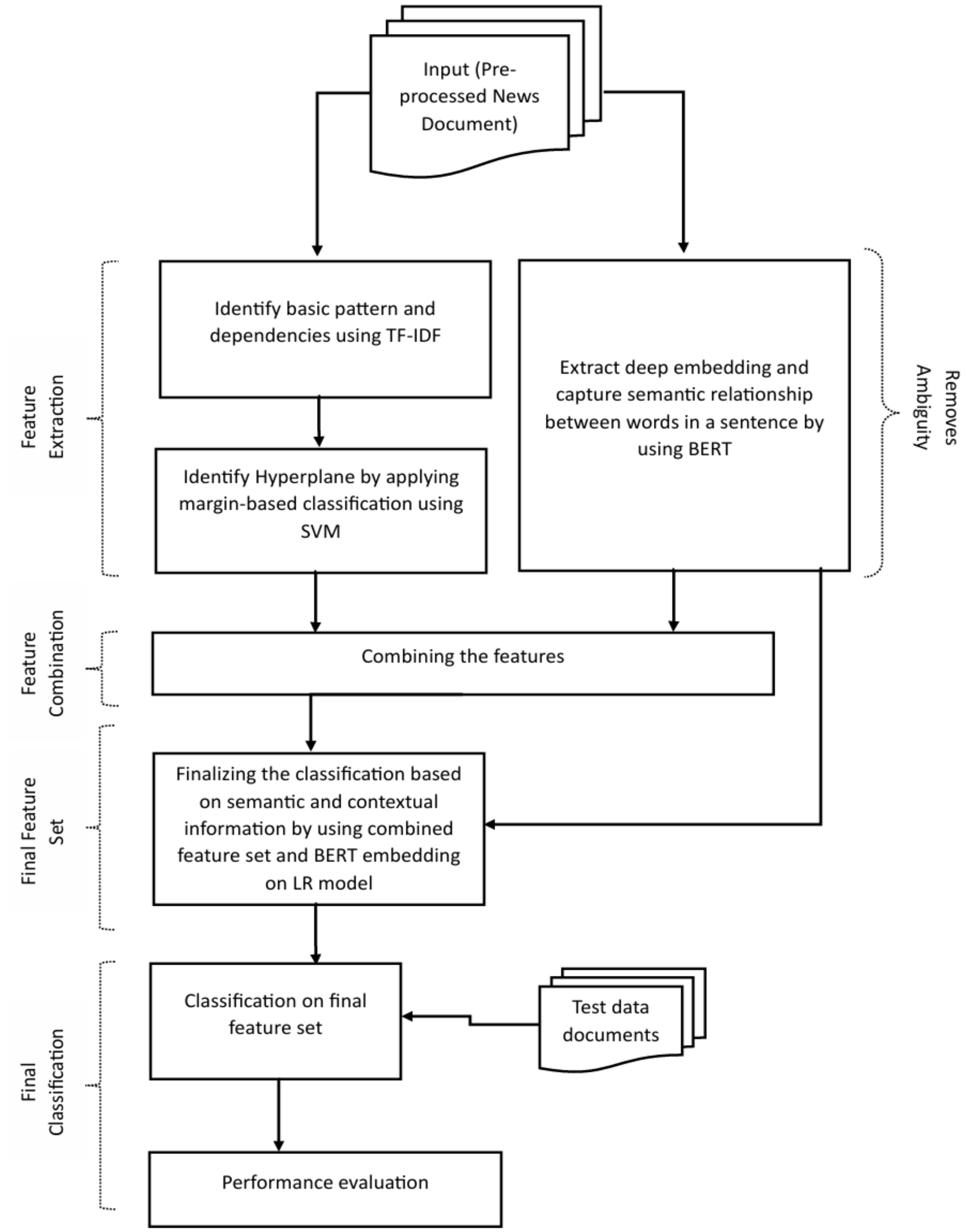
13. Make Predictions on Final Base Model:
- Utilize the final Logistic Regression model feature set to generate prediction for both the training and test data.

14. Calculate Metrics:
- Compute the following metrics:

Accuracy:
$$\text{Accuracy} = (\text{Number of correct predictions}) / (\text{Total number of predictions})$$
Precision:
$$\text{Precision} = (\text{True Positives}) / (\text{True Positives} + \text{False Positives})$$
Recall:
$$\text{Recall} = (\text{True Positives}) / (\text{True Positives} + \text{False Negatives})$$
F1-Score:
$$\text{F1} - \text{Score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$
- Compute the training and validation loss history from the CNN model training.

15. Output Results:
- Print the calculated metrics and loss values to evaluate model performance.

4.5.2 Flow Diagram



Input (Pre-processed News Document)
Identify basic pattern and dependencies using TF-IDF
Identify Hyperplane by applying margin-based classification using SVM
Extract deep embedding and capture semantic relationship between words in a sentence by using BERT
Removes Ambiguity
Feature Extraction
Combining the features
Feature Combination
Finalizing the classification based on semantic and contextual information by using combined feature set and BERT embedding on LR model
Final Feature Set
Classification on final feature set
Test data documents
Performance evaluation
Final Classification

Figure 4.6 Flow Diagram of SBL

4.6 SUMMARY

Chapter 4, Methods and materials we discuss the methods implemented to carry out the various activities in the dataset cleaning and preprocessing stage and analyze the machine learning model for classifying Hindi newswire articles. In this chapter, in section 4.2 we discuss about dataset preparation which includes text preprocessing and feature extraction, in section 4.3 implementation of various hybrid classifier models (combination of three layer model in different sequences) are illustrated, in section 4.4 and 4.5 two best performing or proposed hybrid models are implemented along with its algorithm and flow diagram.

CHAPTER 5
RESULTS AND DISCUSSION

5.1 INTRODUCTION

In any research work, results and discussions are very crucial. In chapter 4 we already discuss the implementation methods of various three layer models. In this chapter, we are going to evaluate the performance of multiclass hybrid models and to analyse a well-structured hybrid approach that appears to be best, considering its high accuracy and less losses in terms of accuracy, precision, F1 score, and recall, achieving a better accuracy rate. The results also highlight that consistency between the size of the training set, the number of classes, and effective data preparation are crucial factors influencing the technique's effectiveness.

5.2 COMPARATIVELY ANALYSIS

The balance between computing efficiency, simplicity, and informativeness is taken into consideration as several hybrid classifier models are examined and contrasted using important metrics such as accuracy, F1 score, recall, and precision. The ultimate goal was to use these factors to determine and suggest the combinations that performed the best.

CNN, SVM, and LR models are used in various sequences in the first combination. The output table below shows all of the important matrices.

Hybrid Model	Train: Test	Epoch	Test Accuracy	Train Accuracy	Validation Loss	Training loss	F1 Score	Recall	Precision
CNN-SVM-LR	80-20	10	61.11	98.91	4.629	4.487	60.40	61.11	62.83
CNN-LR-SVM			61.11	98.99	5.687	4.487	61.28	61.11	63.66
SVM-CNN-LR			60.66	98.84	4.605	4.596	60.07	60.66	62.40
LR-CNN-SVM			49.55	62.84	4.542	3.775	42.42	49.55	51.26

Table 5.1 Output of CLS

Above table indicates that CNN-LR-SVM model is the most effective model since it balances Recall, Precision, and F1 Score while achieving the best test accuracy for both the weighted and macro average across the four models.

Bert, SVM, and LR models are used in various sequences in the second combination. The output table below shows all of the important matrices.

Hybrid Model	**Train: Test**	**Epoch**	**Test Accuracy**	**Train Accuracy**	**F1 Score**	**Recall**	**Precision**
Bert-SVM-LR	80-20	10	59.61	99.40	58.39	59.16	62.20
Bert-LR-SVM			56.76	67.45	53.36	56.76	55.64
SVM-Bert-LR			61.11	98.87	60.87	61.11	62.17
LR-SVM-Bert			60.96	98.84	60.96	60.96	61.85

Table 5.2 Output of SBL

Above table indicates that SVM-Bert-LR model is the most effective model since it balances Recall, Precision, and F1 Score while achieving the best test accuracy for both the weighted and macro average across the four models.

5.3 ANALYSIS OF CNN-LR-SVM

Combining CNN features with TF-IDF features provides a more comprehensive representation of the textual data by capturing both shallow statistical relationships (TF-IDF) and deep semantic patterns (CNN). Here's why this combination can be beneficial:

1. Complementary Information:

   TF-IDF Features: TF-IDF captures word frequency-based features. It focuses on the importance of terms across documents by assigning higher weights to words that are important but not too common. These features represent more traditional, sparse, and statistical patterns, such as the occurrence of specific words that may correlate strongly with certain classes. It provides a bag-of-words-like feature representation, which is useful for identifying key terms, but it does not capture word order or semantic meaning.

   CNN Features: CNNs are good at extracting higher-level patterns from sequences by capturing local dependencies and spatial hierarchies through convolutional filters. The convolutional layers identify patterns, word orders, and dependencies that TF-IDF would miss, such as how different word combinations or phrases contribute to a label. CNNs help capture semantic and syntactic features from text (e.g., meaning derived from context).

2. Combining Shallow and Deep Features:

   Shallow Features (TF-IDF): While TF-IDF focuses on shallow patterns and word occurrences, it may miss the complex relationships between words that exist across the sequence.

   Deep Features (CNN): CNNs can capture deeper relationships and semantic nuances between words in a sentence, but on their own, they may not emphasize the most frequent or statistically important words as TF-IDF does.

   Combining the two can leverage both perspectives: the statistical power of TF-IDF and the

semantic insights of CNNs, leading to better overall feature representation.
In summary, the combination of CNN features (semantic and context-sensitive) and TF-IDF features (statistical and word-frequency based) provides a broader and more powerful representation of the text data. This approach is designed to capture both word-level importance and sequence-based patterns, leading to a more robust classification model.

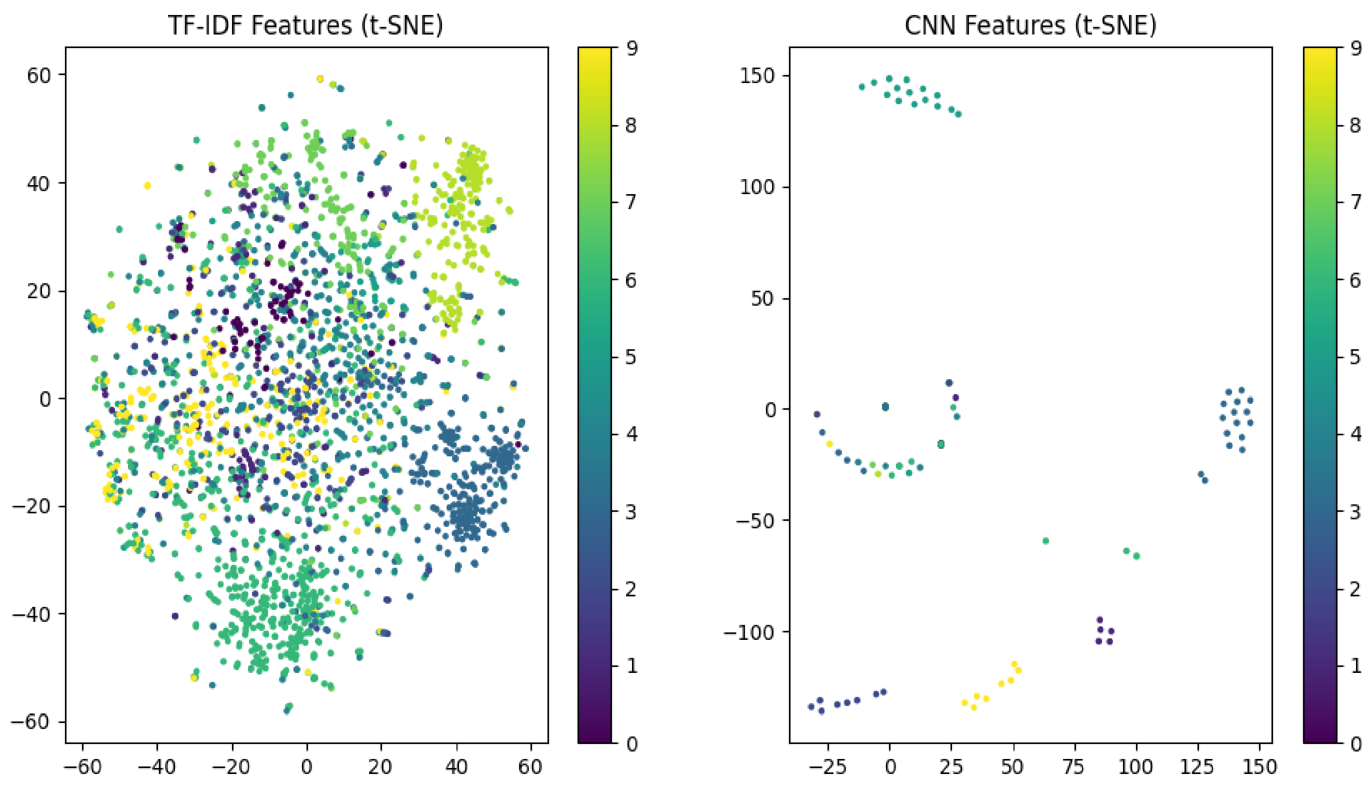


Figure 5.1 Visualization of the TF-IDF features and CNN features

TF-IDF Features (left plot): The points seem to be more spread out and mixed. This is because TF-IDF captures word-level frequency and importance, which leads to a dense set of features. As a result, words from different categories might share similar word frequencies, leading to overlapping clusters.

CNN Features (right plot): The points appear to be more separated into distinct clusters. CNN captures more complex relationships in the text, such as patterns in word sequences, context, and local dependencies. As a result, it can group text data that have similar semantic and syntactic structures, leading to well-separated clusters.

This visualization highlights that **TF-IDF** focuses on word importance without necessarily capturing contextual or relational information, while **CNN** identifies more nuanced and meaningful groupings based on the sequence of words and their context.

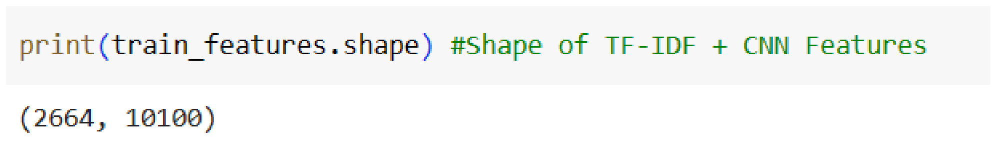
The CNN features' more distinct clustering indicates that it may have learned representations that better differentiate between the categories in your dataset.

Advantages of Combining TF-IDF and CNN Features:

- By combining both **TF-IDF** and **CNN** features, you leverage the strengths of both:
  - **TF-IDF** provides strong word-level frequency importance, which is useful for simple word-based distinctions.
  - **CNN** captures higher-level patterns and context that TF-IDF misses.
- Together, they can improve classification accuracy, especially when your data contains both keyword-based and context-based differences between categories.

Training LR and SVM models on combined features for dimensionality reduction:

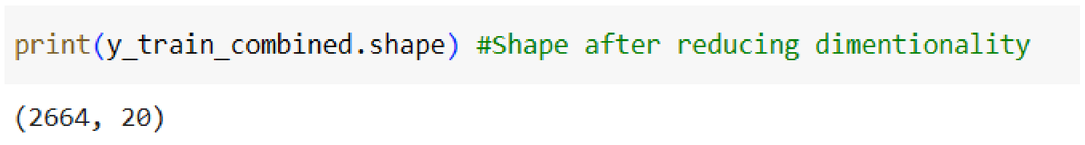
- When you concatenate the **LR** and **SVM probabilities**, the result is a **reduced feature set** compared to the original input feature space
- This reduces dimensionality, making it computationally more efficient to train the final classifier without the risk of overfitting that comes with high-dimensional data.



```
print(train_features.shape) #Shape of TF-IDF + CNN Features
```

```
(2664, 10100)
```

classifier without the risk of overfitting that comes with high-dimensional data.

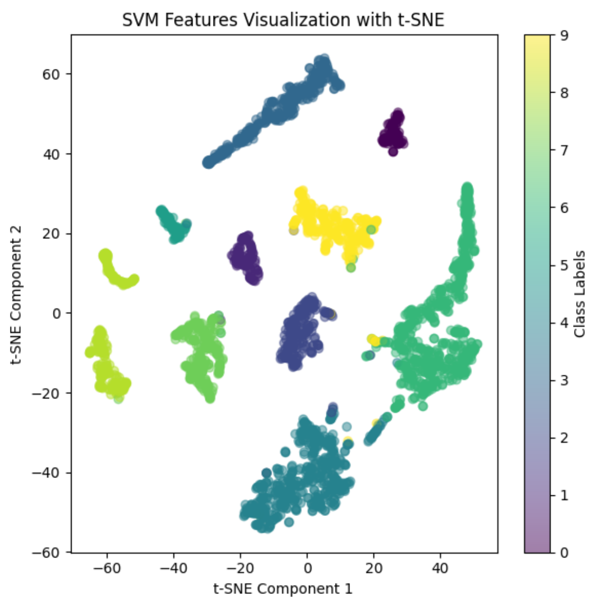


```
print(y_train_combined.shape) #Shape after reducing dimentionality
```

```
(2664, 20)
```

LR and SVM models are trained parallelly because:

- When LR and SVM models are trained independently on the same feature set (combined TF-IDF and CNN features), they each learn to extract and utilize different aspects of the data based on their respective algorithms.
- After training, both models can produce probabilities for each class for the input samples. These probabilities reflect the likelihood of each sample belonging to a particular class, which is a condensed representation of the original feature space.
- By using the output probabilities of LR and SVM as new features for the final classifier, you effectively reduce the dimensionality from the original feature space to a much smaller space (the combined probabilities). This transformation retains the most relevant information for classification while discarding less informative features.
- This output combination can emphasize the strongest features (as determined by the two different models) while reducing the noise introduced by less significant features.



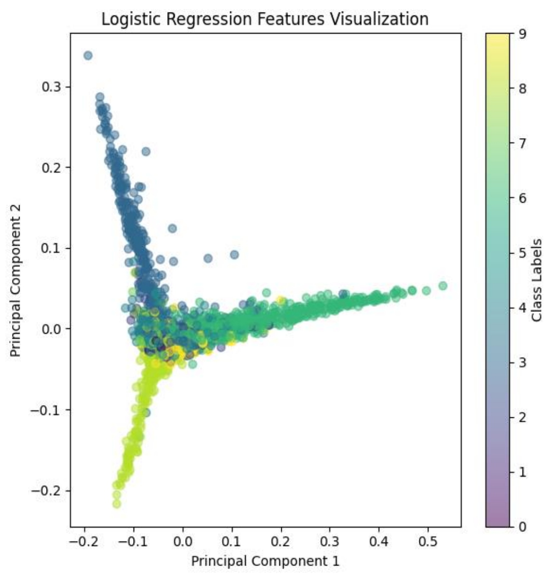


Figure 5.2 Feature Visualization

LR Plot(left):

- The graph has a distinct V-shape, which could reflect how the Logistic Regression model is separating the data into two dominant directions of variance. The sharp angle between the two branches indicates that the model is likely treating some classes in a strongly linear fashion.
- Points at the intersection of the branches could belong to classes that are harder to separate, or where Logistic Regression assigns lower probabilities to specific classes, suggesting potential ambiguity in classification.

SVM plot(right):

- The graph presents several well-separated clusters, each represented by a different color corresponding to distinct class labels (as shown by the color bar on the right). The separations between clusters are indicative of how well the SVM model is distinguishing between classes.
- Some clusters overlap slightly, which may point to some confusion between the corresponding classes. This could mean that the SVM struggles to fully separate these classes in the feature space, possibly due to overlapping data distributions or complexity in those specific classes.

SVM appears to capture more intricate relationships in the feature space, which results in better class separation.
In the **Logistic Regression**, you can see that the data forms a distinct V-shape, which suggests that the underlying features may have some linear separability. Logistic Regression is inherently a linear model, which is beneficial when the data has a linear structure, making it easier to draw simple decision boundaries.

Training the Final Model:

The final model is strategically designed to combine the strengths of both Logistic Regression (LR) and Support Vector Machine (SVM) models for improved classification performance. The choice of SVM as the final base model is supported by its superior ability to distinguish between classes, as evidenced by the visualization and analysis of the model outputs.

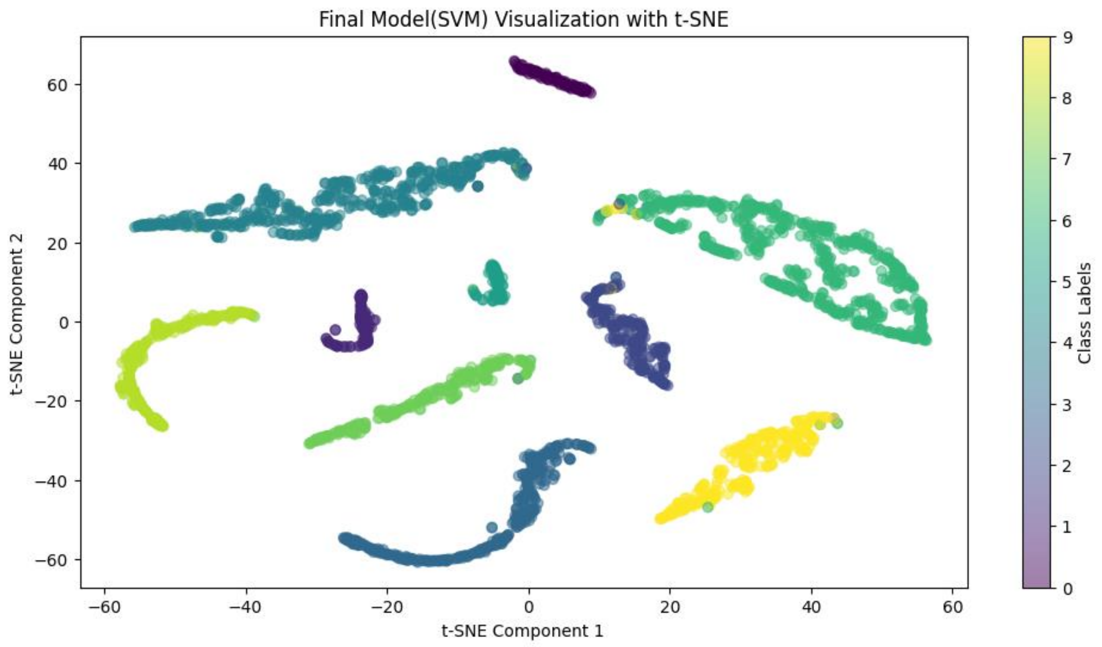


Figure 5.3 Visualization with t-SNE

C lass Cluster:

Distinct Clusters:

- The classes appear well-separated into distinct clusters, which is a good indication that the final SVM model has successfully learned to differentiate between the different class labels.
- Some classes like the dark purple (class 8) and yellow (class 9) are particularly well-separated from other clusters, indicating strong decision boundaries for these classes.

Shape of Clusters:

- The **elongated shapes** of some clusters (like the teal cluster for class 6 or the yellow cluster for class 9) indicate that the features may be more complex or exhibit variations within the class. The model seems to have captured these internal variations within the class boundaries.
- Other clusters, such as the green cluster (class 3), are more compact, which could suggest more uniformity in the features for those particular classes.

Comparison with Previous SVM t-SNE Visualization:

- Compared to the earlier SVM visualization (in the first graph you shared), this final SVM t-SNE plot shows even more distinct and clearly defined clusters. This suggests that the final model has undergone improvements, potentially through fine-tuning or a more refined combination of features.
- The previous SVM plot had some overlap between classes, while the final model shows better separation and less confusion between class boundaries.

The visualization of the final SVM model shows clear, well-separated class clusters with some minor overlaps. This indicates that the model has successfully captured the key distinctions between most classes, making it well-suited for classification tasks.

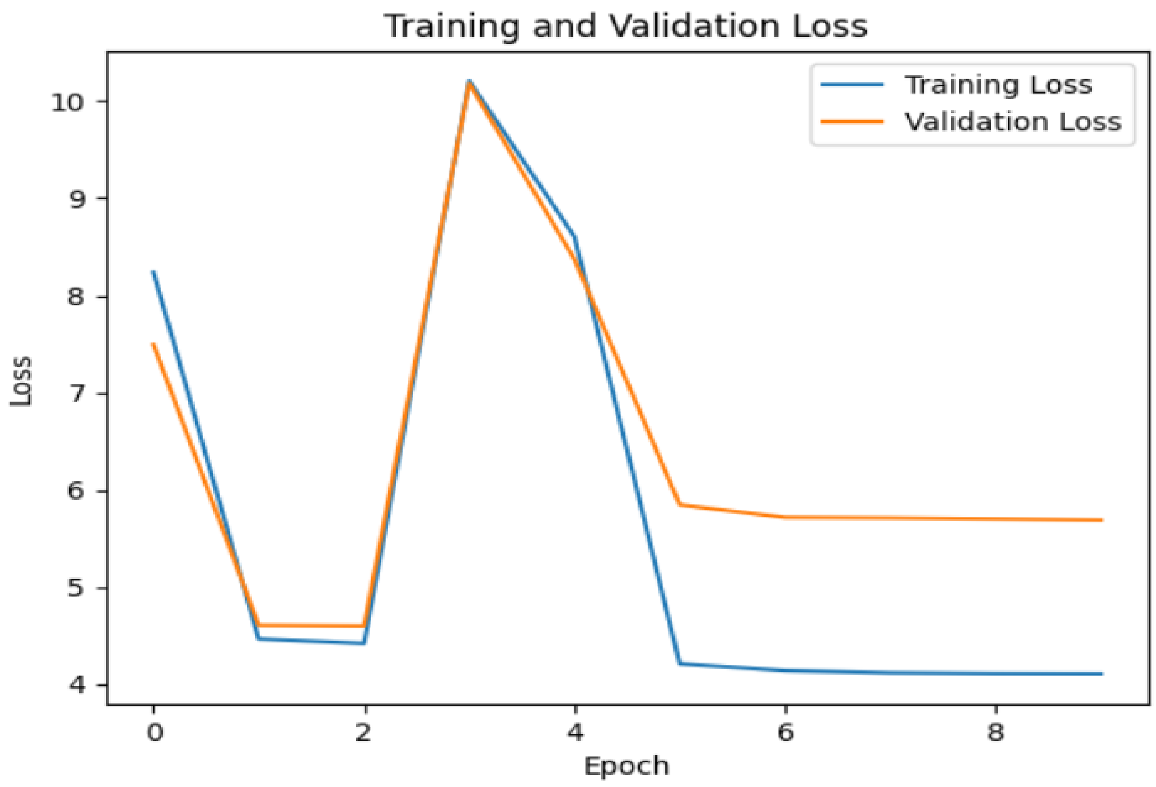
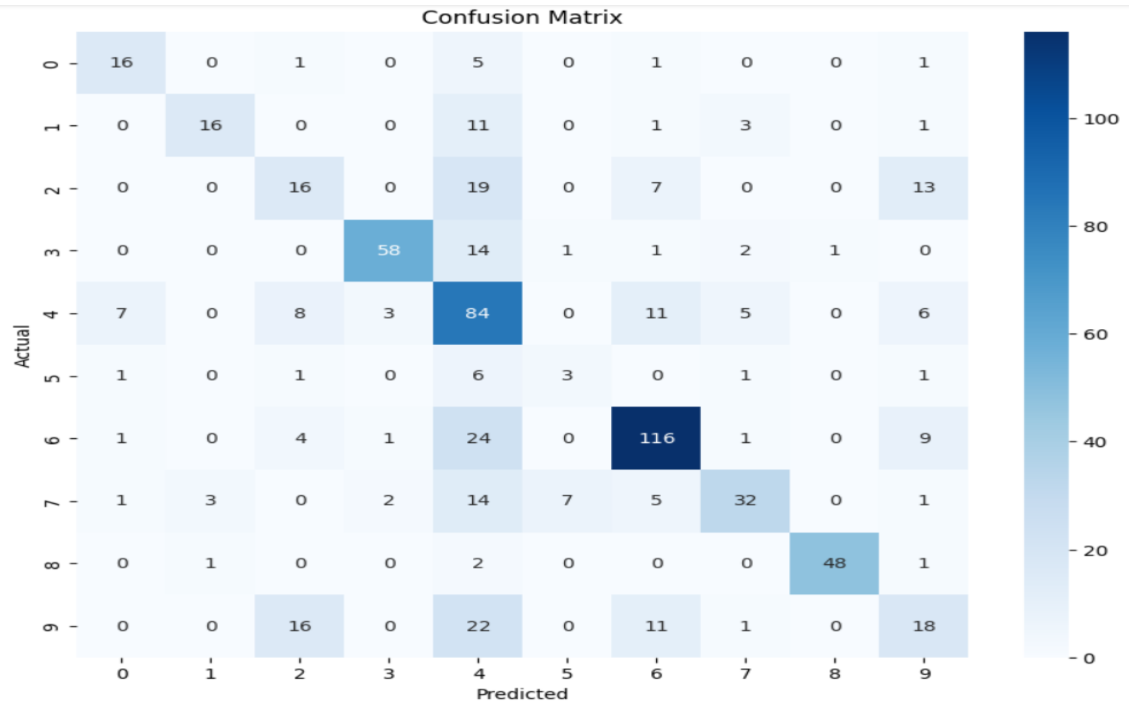


Figure 5.4 Loss Graph of CLS model



	0	1	2	3	4	5	6	7	8	9
0	16	0	1	0	5	0	1	0	0	1
1	0	16	0	0	11	0	1	3	0	1
2	0	0	16	0	19	0	7	0	0	13
3	0	0	0	58	14	1	1	2	1	0
4	7	0	8	3	84	0	11	5	0	6
5	1	0	1	0	6	3	0	1	0	1
6	1	0	4	1	24	0	116	1	0	9
7	1	3	0	2	14	7	5	32	0	1
8	0	1	0	0	2	0	0	0	48	1
9	0	0	16	0	22	0	11	1	0	18

Figure 5.5 Confusion Matrix of CLS model

5.4 ANALYSIS OF SVM-Bert-LR

The hybrid model processes the dataset through three main layers:

1. **TF-IDF Vectorization [Layer 0]**: Converts text into numerical vectors based on word importance.
2. **SVM Layer [Layer 01]**: Performs dimensionality reduction (via PCA) and learns a margin-based separation in the feature space.
3. **BERT + Logistic Regression [Layer 02 & Layer 03]**: Incorporates semantic understanding using BERT embeddings and classifies data with Logistic Regression.

We will analyse the transformations at each stage using the four PCA-reduced 2D visualizations of the dataset provided earlier.

Analysis of Each Layer and Its Contribution

1. TF-IDF Layer [Layer 0]

Code Explanation:

- TF-IDF vectorizer converts the textual data into numerical vectors, assigning weights to words based on term frequency (TF) and inverse document frequency (IDF).
- This stage captures **lexical differences** but lacks deeper semantic understanding.

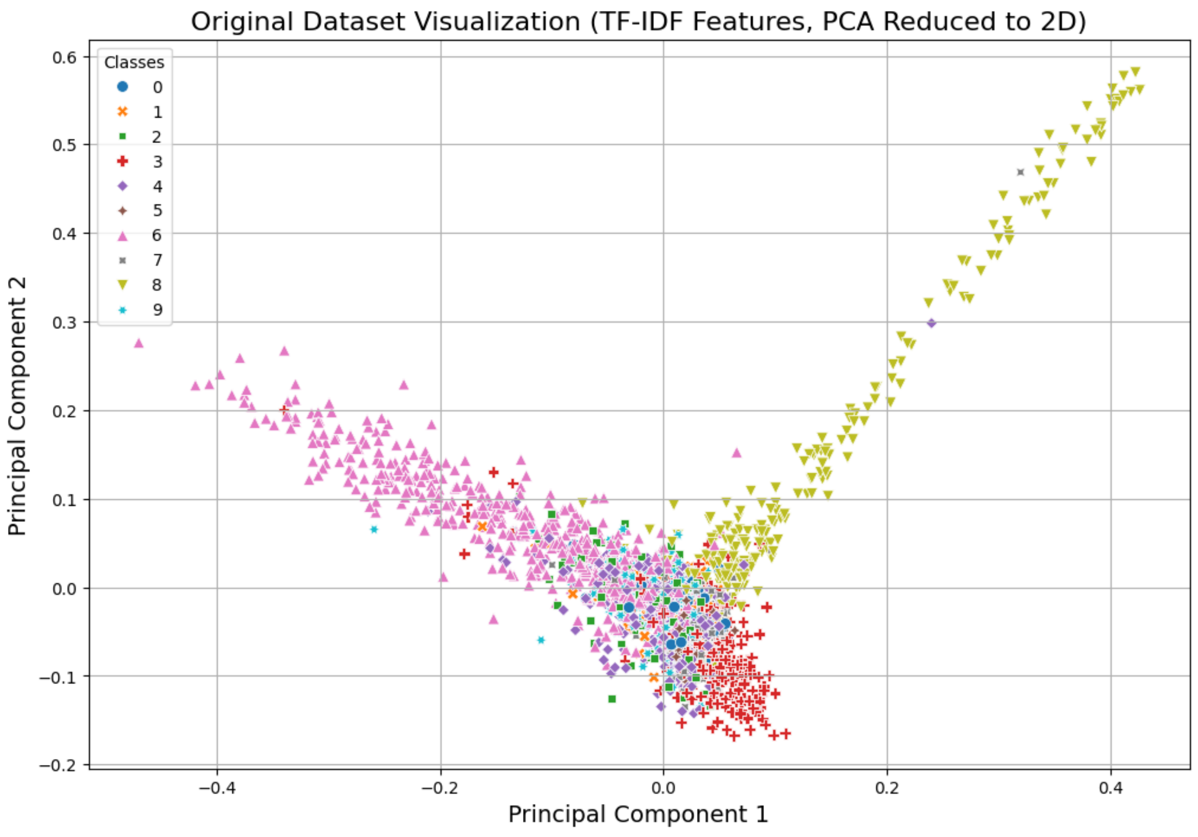


Figure 5.6 Dataset Visualization

- **Graph Analysis**

  Graph 1 (Post-TF-IDF + PCA):

  - The points likely form loose clusters with overlapping boundaries.
  - The separation between classes is weak, as TF-IDF primarily focuses on surface-level features (e.g., word counts and patterns) and doesn't capture contextual relationships between words.

Insights:

- At this stage, the model identifies basic patterns and dependencies but struggles with nuanced classification.

2. SVM Layer [Layer 1]

- **Code Explanation**:
  - SVM applies margin-based classification to identify hyperplanes that separate classes in the TF-IDF vector space.
  - PCA is applied for dimensionality reduction, improving visualization and reducing feature redundancy.

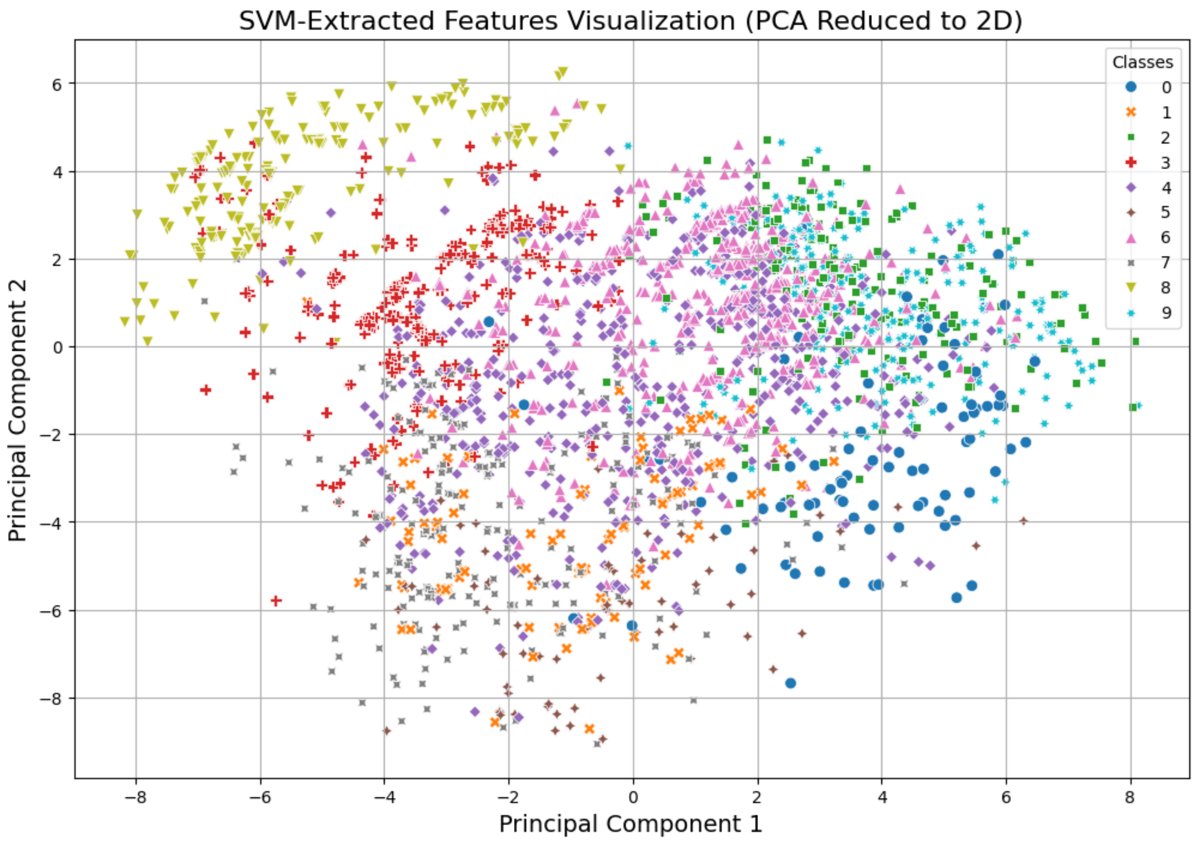


Figure 5.7 SVM Extracted Feature

- **Graph 2 (Post-SVM + PCA)**:
  Noticeably improved separation between classes compared to Graph 1.
  SVM attempts to draw linear boundaries, leading to tighter clusters and clearer differentiation in the projected 2D space.
  Overlap still exists in more complex or semantically similar classes, reflecting SVM's reliance on TF-IDF features.

- **Insights**:
  SVM enhances class separability by leveraging high-dimensional hyperplanes, but its reliance on shallow features limits its ability to fully resolve ambiguities.

3. BERT + Logistic Regression Layer [Layer 2 & 3]

Code Explanation:

- BERT generates contextualized embeddings, capturing semantic relationships between words in a sentence.

- Logistic Regression performs final classification on these embeddings.

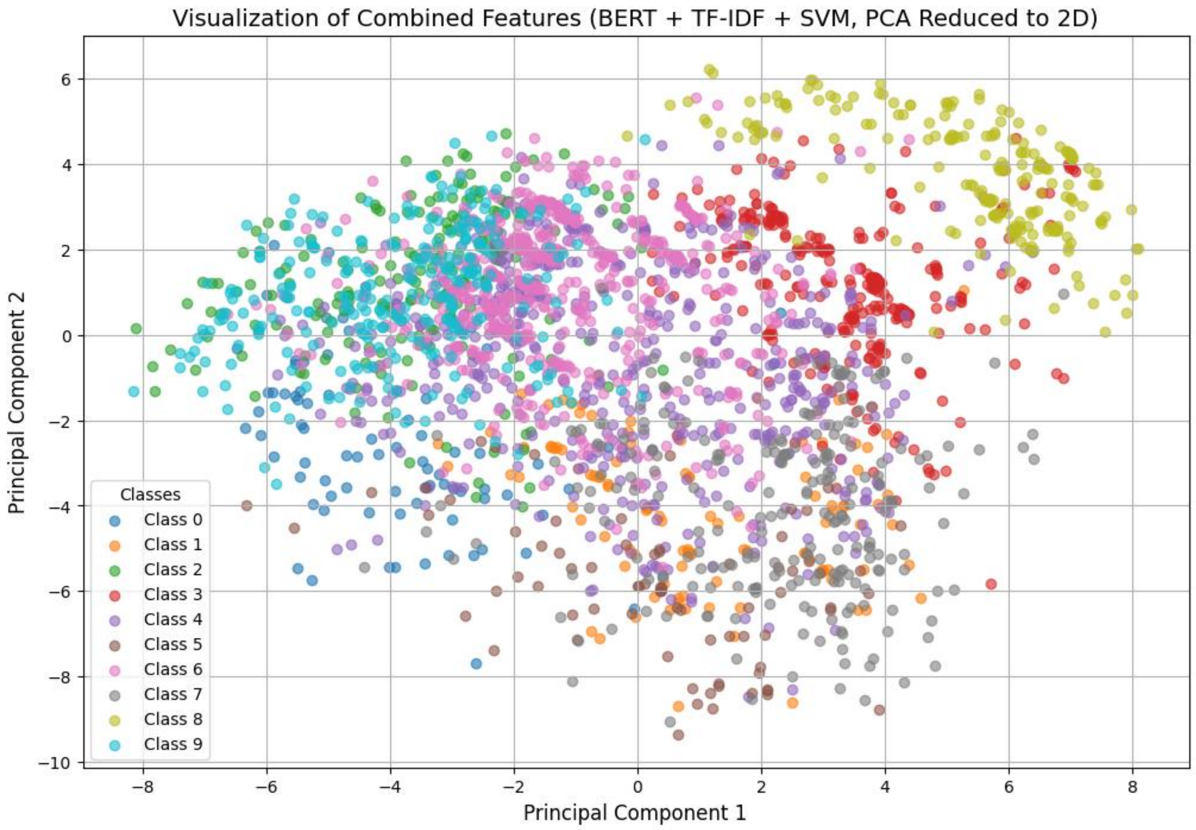


Figure 5.8 Visualization of combined feature

- **Graph Analysis**:
  Graph 3 (Post-BERT + PCA) [Layer 2]:
  - The clusters become more distinct, with reduced overlap.
  - BERT embeddings encode deep semantic information, helping resolve ambiguities from the TF-IDF stage (e.g., polysemy and synonymy issues).

  Graph 4 (Post-BERT + Logistic Regression + PCA) [Layer 3]:
  - Final separation is the clearest, with tightly grouped clusters and minimal overlap.
  - Logistic Regression utilizes the BERT embeddings effectively, finalizing the classification based on semantic and contextual information.

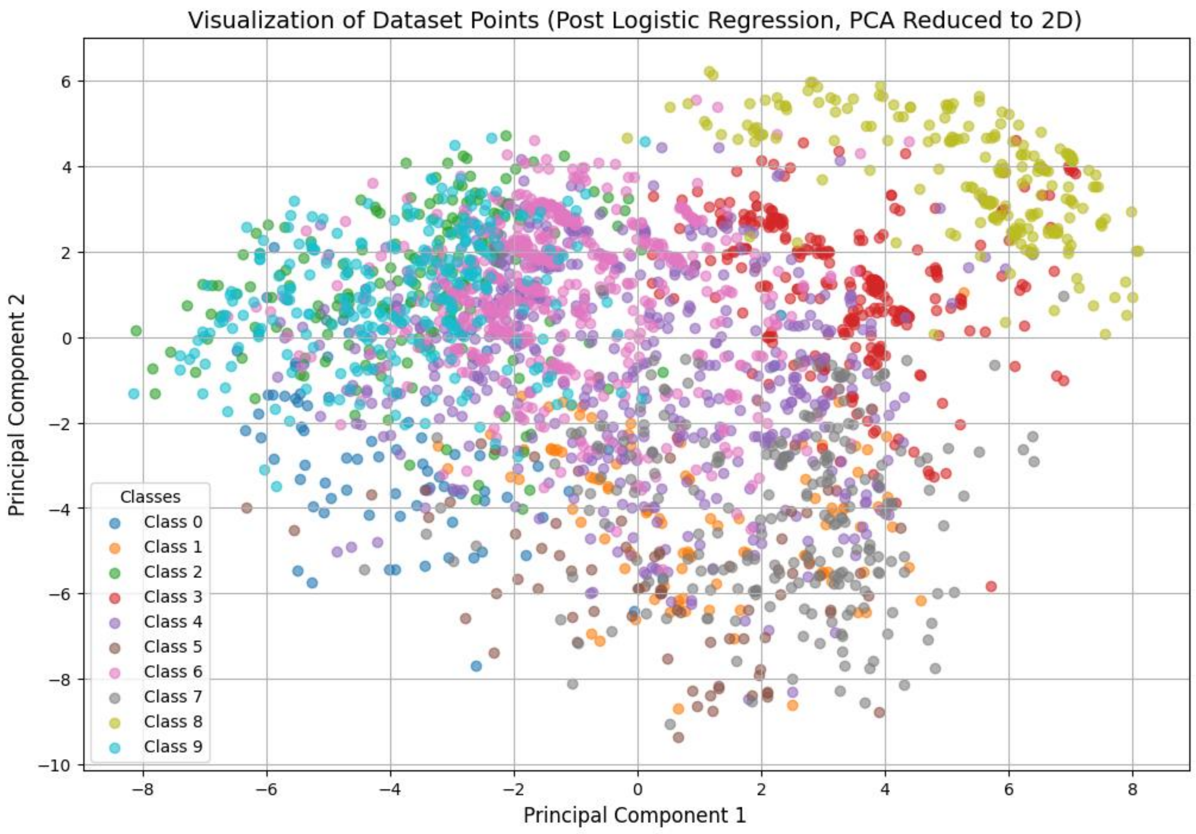


Figure 5.9: Visualization of Dataset Points

- **Insights**:
    - BERT embeddings provide a substantial improvement in feature representation, and Logistic Regression further optimizes decision boundaries in this enhanced feature space.

Overall Analysis of Dataset Transformation

1. **TF-IDF Layer**:
   Focuses on word-level statistical features.
   Provides a basic separation but lacks deeper semantic insights, leading to overlapping clusters.
2. **SVM Layer**:
   Builds upon TF-IDF features by learning linear hyperplanes to separate classes.
   Improves class separability but struggles with complex, contextual dependencies.
3. **BERT + Logistic Regression Layer**:
   Introduces contextualized embeddings that capture word meanings in relation to their surroundings.
   Logistic Regression enhances final separation, yielding tight clusters and reducing errors caused by ambiguity.

Key Observations

- Each successive layer reduces overlap between class clusters by introducing more sophisticated feature representations.
- The transition from shallow (TF-IDF) to deep (BERT) features demonstrates a clear improvement in class separability.
- PCA projections validate the effectiveness of each layer, showing progressive refinement in how the dataset points are represented.

This layered approach combines the strengths of TF-IDF's statistical features, SVM's margin-based separation, and BERT's semantic richness, culminating in a robust classification pipeline.

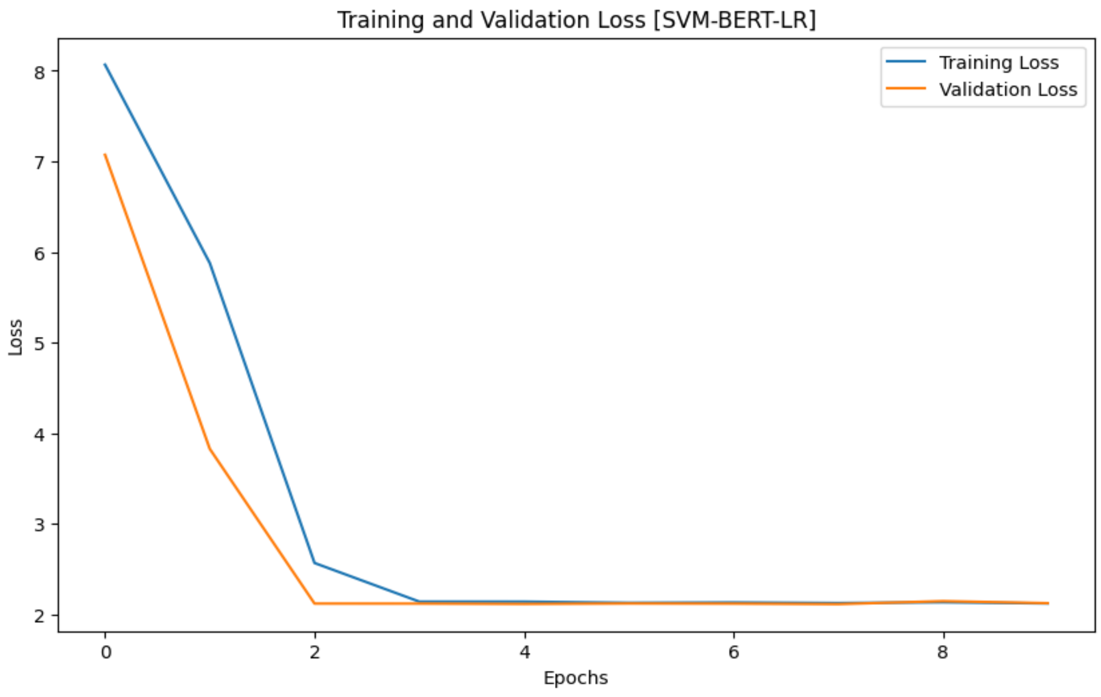
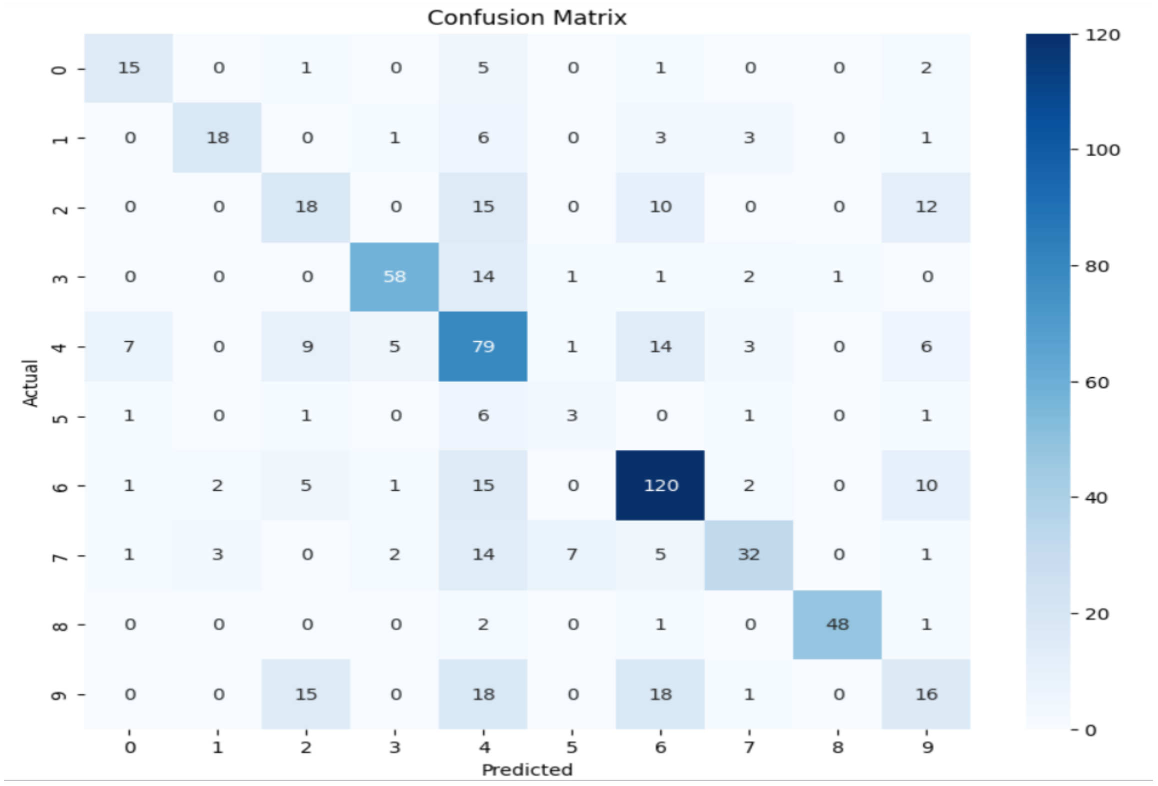


Figure 5.10: Loss graph of SBL model



Actual \ Predicted	0	1	2	3	4	5	6	7	8	9
0	15	0	1	0	5	0	1	0	0	2
1	0	18	0	1	6	0	3	3	0	1
2	0	0	18	0	15	0	10	0	0	12
3	0	0	0	58	14	1	1	2	1	0
4	7	0	9	5	79	1	14	3	0	6
5	1	0	1	0	6	3	0	1	0	1
6	1	2	5	1	15	0	120	2	0	10
7	1	3	0	2	14	7	5	32	0	1
8	0	0	0	0	2	0	1	0	48	1
9	0	0	15	0	18	0	18	1	0	16

Figure 5.11 Confusion matrix of SBL model

5.5 SUMMARY

Precision, Recall, and F1 Score are essential parameters to judge the system's performance. Precision is the number of correctly translated words to output length; Recall is the number of correctly translated words to reference length; F1 score is the harmonic mean of precision and recall. CNN-LR-SVM model and SVM-Bert-LR model is the most effective model since it balances Recall, Precision, and F1 Score while achieving the best test accuracy for both the weighted and macro average across the four models.

CHAPTER 6
SUMMARY AND CONCLUSION

6.1 INTRODUCTION

As discussed earlier, the entire research is arranged and documented in different chapters.In Chapter 1, a brief introduction to the research work is presented. Chapter 2 discusses and literature available in a similar area of the research. Based on that literature, the objectives of the research work are identified. Chapter 3 is about the theoretical analysis of the machine learning and deep learning models, and how ML integrate with NLP for better classification. All substrate materials used in the research are discussed in Chapter 4. After that, in the same chapter, the methodology for developing hindi news classification system using various models. Chapter 5 discusses the results of theproposed two models for classification system and its performance in terms of accuracy. Outline of proposed work and the future scope, are discussed in sections 6.2 and 6.3 respectively. Finally, the chapter is summarized in section 6.4.

6.2 OUTLINE OF PROPOSED WORK

Text classification is particularly vital due to the abundance of uncategorized data. The surge in content on Hindi news platforms calls for better automated methods to organize and access it. This study aims to instigate Deep learning, Natural Language Processing and Machine Learning approaches to classify Hindi news articles efficiently. In this study we suggest two models with a well-structured hybrid approach that appears to be best, considering its high accuracy and less losses.

Combining CNN-SVM-LR model into a single pipeline for text categorization leverages the strengths of both deep learning and classical machine learning models. It manages complex decision boundaries, effectively categorize, and capture local attributes. When deep learning techniques are insufficient for final classification refinement or when traditional models may have trouble capturing complex patterns, this hybrid model can be especially useful. It involves feature combination and TF-IDF vectorization.

Combining SVM-BERT-LR model, blends the advance developments in deep learning with reliable traditional machine learning approaches. Utilizing the contextual embeddings of BERT, the classification efficiency of LR, and the decision-refinement power of SVM, this model provides a comprehensive solution for complex text categorization tasks.

6.3 FUTURE SCOPE OF WORK

The research work inspires future research directions and potential extensions. The development of Hindi news classification models has vast potential, especially with the increasing demand for regional language content and advancements in machine learning (ML) and natural language processing (NLP). The different directions described that may explore future research works are:

- Develop models capable of classifying news in Hindi-English mixed text (Hinglish) to cater to the growing prevalence of mixed-language usage in digital platforms.
- Move beyond generic categories (e.g., sports, politics) to subcategories like cricket, football, or elections.
- Create specialized models for niche domains like agriculture, regional development, or social issues.
- Focus on tagging articles related to specific types of events, such as natural disasters, economic trends, or festivals.
- Use classification models to deliver personalized news recommendations based on user interests, sentiment preferences, or geographic location.
- Combine classification models with user behavior analysis to predict intent and suggest relevant content.

6.4 SUMMARY

This chapter discusses the summary of the entire research work. In the introduction section of the chapter, a brief overview of all the chapters is presented. After that outline of proposed work and the future scope of this research work is discussed, leading to the future extension of the work carried out in this research.

REFERENCES

[A]. Jain, V., & Kashyap, K. L. (2023). "Ensemble hybrid model for Hindi COVID-19 text classification with metaheuristic optimization algorithm". Multimedia Tools and Applications 82:16839–16859 https://doi.org/10.1007/s11042-022-13937-2.

[B]. Krishnamoorthy, P., Sathiyanarayanan, M., & Proença, H. P. (2024). A novel and secured email classification and emotion detection using hybrid deep neural network. International Journal of Cognitive Computing in Engineering. https://doi.org/10.1016/j.ijcce.2024.01.002.

[C]. Dodda, R., & Alladi, S. B. (2024). Enhancing document clustering with hybrid recurrent neural networks and autoencoders: A robust approach for effective semantic organization of large textual datasets. *EAI Endorsed Transactions on Intelligent Systems and Machine Learning Applications, 1*(1).

[D]. Livieris, I.E., Iliadis, L., & Pintelas, P. (2020). On ensemble techniques of weight-constrained neural networks. *Evolutionary Systems*, 1–13. https://doi.org/10.1007/s12530-019-09331-w .

[E]. Mohammed, A., & Kora, R. (2022). An effective ensemble deep learning framework for text classification. *Journal of King Saud University – Computer and Information Sciences, 34*, 8825–8837. https://doi.org/10.1016/j.jksuci.2021.11.001

[F]. Sahoo Kumar Sovan, Saha Saumajit, Ekbal Asif, Bhattacharyya Pushpak and Mathew Jimson (2019) "Event-Argument Linking in Hindi for Information Extraction in Disaster Domain" *CICLing 2019*.

[G]. Ahmad Zishan, Sahoo Kumar Sovan, Ekbal Asif, Bhattacharyya Pushpak (2018) "A Deep Learning Model for Event Extraction and Classification in Hindi for Disaster Domain" *Proc. of ICON*-2018, Patiala, India. December 2018 c2018 NLPAI, pages 127–136.

[H]. Shah, K., Patel, H., Sanghvi, D. et al. (2020). A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classification. Augment Hum Res 5, 12 (2020). https://doi.org/10.1007/s41133-020-00032-0.

[I]. H. Alzoubi, Yehia Ibrahim, Ahmet E. Topcu, and Ahmed Enis Erkaya. (2023). "Machine Learning-Based Text Classification Comparison: Turkish Language Context" *Applied Sciences* 13, no. 16: 9428. https://doi.org/10.3390/app13169428.

[J]. Mccallum A, Nigam K (2001) A comparison of event models for naive bayes text classification. Work LearnText Categ752:05

[K]. Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative Study of CNN and RNN for Natural Language Processing. *ArXiv*. /abs/1702.01923.

[L]. Jamshidi, S., Mohammadi, M., Bagheri, S., Esmaeili Najafabadi, H., Rezvanian, A., Gheisari, M., Ghaderzadeh, M., Shahabi, A. S., & Wu, Z. (2024). Effective text classification using BERT, MTM LSTM, and DT. *Data & Knowledge Engineering, 151,* 102306. https://doi.org/10.1016/j.datak.2023.102306

[M]. Sitaula, C., Shahi, T.B. (2024). Multi-channel CNN to classify Nepali COVID-19 related tweets using hybrid features. J Ambient Intell Human Comput 15, 2047–2056 (2024). https://doi.org/10.1007/s12652-023-04692-9

[N]. Jain, V., Kashyap, K.L. (2023). Ensemble hybrid model for Hindi COVID-19 text classification with metaheuristic optimization algorithm. Multimed Tools Appl 82, 16839–16859 (2023). https://doi.org/10.1007/s11042-022-13937-2

[O]. Mundra, S., Mittal, N. (2022). FA-Net: fused attention-based network for Hindi English code-mixed offensive text classification. Soc. Netw. Anal. Min. 12, 100 (2022). https://doi.org/10.1007/s13278-022-00929-1