

Thesis submitted to the
UNIVERSITY OF MOHAMED BOUDIAF – MSILA



FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in Computer science

By
Oucif Ibtissam
Sahnoune Omayma Melek

Title of the thesis

Sentiment Analysis in Social Media Posts using Deep Learning

Under the supervision of **Bilal**
Lounnas

Composition of the jury

Nourelhouda Chalabi	University of Msila	President
Bilal Lounnas	University of Msila	Reporter
Marouane Kihal	University of Msila	Examiner

JUNE, 2026

Abstract

This master's thesis presents a rigorous empirical comparative study evaluating text sentiment classification capabilities of deep learning and Transformer-based architectures. Utilizing two benchmark datasets, the SST-2 binary sentiment corpus [29] and the TweetEval sentiment dataset [8], the research designs, implements, and optimizes three distinct neural pipelines: CNN, LSTM, and fine-tuned DistilBERT. The empirical results reveal that finetuned DistilBERT achieves the highest performance in sentiment classification with a testing accuracy of 70.29% on TweetEval, while the LSTM network records a testing accuracy of 56.38%. CNN achieves competitive results with a testing accuracy of 57.95%. Furthermore, an independent cross-domain validation experiment tracking 1,500 ChatGPT-generated sentences [15] reveals distinct architectural biases under distribution shifts. The study addresses critical technical challenges including severe class imbalance and hardware optimization constraints, and proposes future research directions toward integrating Large Language Models (LLMs) and extending these pipelines to Algerian Dialectal Arabic (Darja).

المخلص

تقدم أطروحة الماستر هذه دراسة مقارنة تجريبية صارمة لتقييم قدرات تصنيف مشاعر النصوص باستخدام بنى التعلم العميق ونماذج المحولات. بالاعتماد على مجموعتي بيانات قياسيتين SST-2 و TweetEval، البحث في تصميم وتنفيذ وتحسين ثلاثة مسارات عصبية LSTM، CNN ونموذج DistilBERT المضبوط بدقة. تكشف النتائج التجريبية أن DistilBERT حقق أعلى أداء بدقة اختبار 70.29% على TweetEval، فيما سجلت LSTM دقة 56.38%، وحققت CNN نتائج تنافسية بدقة 57.95%. علاوةً على ذلك، أظهرت تجربة تحقق مستقلة تتبعت 1,500 جملة توليدية تحيزات معمارية مميزة تحت تحولات التوزيع. وتخلص الدراسة إلى تحديد التحديات التقنية الحرجة كالاختلال في توازن الفئات، وتطرح مسارات بحثية مستقبلية نحو دمج نماذج اللغة الضخمة وتوسيع هذه المسارات لتشمل اللهجة الجزائرية الدارجة.

Dedications:

" I dedicate this achievement to my beloved parents, MASOUD & RABIHA, for their endless love and support. To the pure soul of my late sister, SAMIA, whose memory lives forever in my heart. To my sister, brothers, and my dear friend and partner, MELEK, for sharing this journey.

To my deep passion and the future I am building step by step with resilience."

OUCIF Ibtissam

" To those who lit up my path and supported me in difficult times...

I dedicate this graduation to the source of love and my safe place (My Dear Mother), and to the one who taught me that ambition has no limits and success needs patience (My Dear Father).

To my support and great companions; my brothers and sisters, and my loyal friends who shared with me the long nights of study and the joy of success, and to everyone who prayed for me.

To all of you, I dedicate this humble work."

Sahnoune Omayma Melek

Acknowledgments

First and foremost, we owe our deepest gratitude to Allah, the Almighty, the Most Gracious, and the Most Merciful, who granted us the health, strength, cognitive ability, and immense patience to overcome all the challenges and successfully accomplish this research work.

We would like to express our formal and sincere gratitude to our research supervisor, Mr. Bilal Lounnas, for his academic guidance, constructive supervisor oversight, and for dedicating his valuable time and insights to review this project during the preparation of this master's thesis.

Our deep appreciation and highest respect also go to the honorable members of the jury. We are highly grateful to them for accepting the task of evaluating, reviewing, and examining this manuscript, and for providing their constructive academic expertise to enrich the value of this study.

We also wish to extend our special thanks to the Department of Computer Science at the Faculty of Mathematics and Computer Science, University of Mohamed Boudiaf – Msila. We thank all the professors, administrative staff, and educators who guided us throughout our academic years.

Finally, this page would not be complete without expressing our profound gratitude to our beloved families. We thank our parents and families for their continuous love, unconditional sacrifices, ongoing prayers, and endless psychological support, which have been the true fuel behind our perseverance and success.

Contents

List of tables.....	v
Introduction	1
Chapter 1:Text Mining and Sentiment Analysis.....	2
1.1. Introduction:	2
1.2.1. Web 2.0 and Social Data:	3
1.2.2. Text Mining and Knowledge Discovery:.....	4
1.2.3. Text Mining and NLP Relations:.....	4
1.3 Sentiment Analysis:	5
1.3.1. Definition of Sentiment Analysis:.....	5
1.3.2. Levels of Sentiment Analysis:	6
1.4 Sentiment Analysis Challenges:.....	7
1.4.1. Linguistic and Semantic Obstacles:	7
1.4.2. Microblogging Corpora Dynamics:	8
1.5 Sentiment Analysis Applications:	9
1.5.1. Corporate & Market Intelligence:	9
1.5.2. Public Opinion and Polarity:.....	9
1.6. Conclusion:	10
Chapter 2: Deep Learning and Language Models	11
2.1. Introduction:	11
2.2. Neural Network Foundations:.....	11
2.2.1. Perceptron Structure and Architecture:.....	12
2.2.2. Activation and Loss Functions:.....	12
2.2.3. Neural Network Optimization and Training:	13
2.3. Text specific Neural Architectures:	15
2.3.1. Recurrent Neural networks (RNN):	15
2.3.2. Long Short-Term Memory (LSTM):	16
2.3.3. Convolution Neural Networks (CNN):	17
2.4. Transformers and Attention Mechanisms:.....	18
2.4.1. Self-Attention Mechanisms:	18
2.4.2. BERT Architecture:.....	19
2.4.3. DistilBERT Model:	20

2.5. Evaluation Metrics:.....	22
2.5.1. Precision, Recall, and Accuracy:	22
2.5.2. F1-Score and Confusion Matrix:.....	23
2.6. Conclusion:	23
Chapter 3: Technical Methodology and Data Engineering	25
3.1. Introduction:	25
3.2. Dataset Characteristics & Evaluation Framework:.....	25
3.2.1. Benchmark Datasets:	26
3.2.2. External Synthetic Validation Dataset:	26
3.3. Text Processing & Tokenization Engine:	26
3.3.1. Cleaning Pipelines:	26
3.3.2. Tokenization Paradigms:	27
3.3.3. Padding and Attention Masking:	28
3.4. Deep Learning Architectures:	29
3.4.1. DistilBERT (Transformer-based Model):.....	29
3.4.2. Long Short-Term Memory (LSTM) Network:	29
3.4.3. Convolutional Neural Network (CNN):.....	29
3.5. Conclusion:	30
Chapter 4: Implementation, Results, and Discussion	31
4.1. Introduction:	31
4.2. Development Environment and Computational Setup:	31
4.2.1. Development Environment:.....	31
4.2.2. Model Configurations:	32
4.3. Results Analysis and Discussion:.....	32
4.3.1. Stanford Sentiment Treebank 2:	32
4.3.2 TweetEval:.....	36
4.1. Conclusion:	42
Conclusion.....	44

List of figures

Fig. 1.1: The synergistic relationship between NLP and Text Mining.	6
--	---

Fig. 1.2: Granularity levels of Sentiment Analysis..	9
Fig. 2.1: Neural Network Training Loop Diagram (Forward and Backward Pass).	16
Fig. 2.2: RNN Hidden State Transition and Information Flow.	17
Fig. 2.3: Structural flow and gated mechanism within an LSTM cell at time step t.	18
Fig. 2.4: Sequential processing workflow of the Self-Attention mechanism.	21
Fig. 2.5: The two-stage paradigm of BERT: Pre-training on unlabeled text and Fine-tuning on target task data.	22
Fig. 2.6: Knowledge distillation framework compressing Full BERT (Teacher) into DistilBERT (Student)..	23
Fig. 3.1: Sequential architecture of the text sanitization pipeline...	28
Fig. 3.2: Parallel tokenization strategy across the model pipelines....	29
Fig. 4.1: Accuracy and F1-score Evaluation Across Models.....	34
Fig. 4.2: Comparison of Deep Learning performance.....	38
Fig. 4.3: Blind Performance Test on External Tweeter Data	42

List of tables

Table 2.1: Confusion Matrix for Sentiment Classssification.	24
Table 4.1 : Performance Comparison of Different Deep Learning Models.	34
Table 4.2 : Combined confusion Matrix for the three Models (DistilBERT, CNN,and LSTM).	35
Table 4.3: Comparaision of Model Ececution Time (SST-2) .	36
Table 4.4: Performance Comparison of Different Deep Learning Models .	37

Table 4.5: DistilBERT confusion Matrix	39
Table 4.6: CNN confusion Matrix	39
Table 4.7: LSTM confusion Matrix	40
Table 4.8: Comparison of Model Execution Time(TweetEval)	41

Introduction

Over the past decade, the explosion of digital communication and social media has completely changed how people interact and share information. Platforms like X (formerly Twitter) have become massive, fast-moving sources of unstructured text that capture real-time public opinions, emotions, and social trends. Extracting valuable insights from this continuous stream of data is incredibly important for fields ranging from market analysis and brand monitoring to public policy and crisis management.

However, automatically making sense of these texts presents major challenges that traditional Natural Language Processing (NLP) methods struggle to handle. Social media posts are often short, grammatically messy, full of slang, and constantly changing in context. While older, rule-based algorithms like SVM or Naive Bayes are computationally efficient, they rely heavily on manual feature engineering and simply cannot grasp the complex, nonlinear relationships and contextual nuances of human language.

To overcome these limitations, the field has shifted toward Deep Learning models that can automatically learn feature representations. Convolutional Neural Networks (CNNs) became popular for catching local patterns, while Long Short-Term Memory (LSTM) networks helped track sequential context. More recently, Transformer architectures and pre-trained language models like BERT, along with its lighter version DistilBERT, completely changed the game by understanding bidirectional context, achieving unprecedented accuracy in sentiment classification.

In this context, this thesis provides a rigorous empirical comparative study to evaluate how well different neural architectures classify sentiment. We compare traditional deep learning models (CNN and LSTM) with an advanced Transformer model (fine-tuned DistilBERT). To test how robust these models are when data patterns change, we also included a cross-domain validation experiment using 1,500 ChatGPT-generated sentences. The main research question guiding this work is: How do deep learning and Transformer architectures compare in classifying sentiment in short, noisy texts, and what specific biases do they show when faced with data distribution shifts?

To answer this, the thesis is organized into four main chapters:

Chapter 1 lays the theoretical foundations of text mining and sentiment analysis, focusing on the unique characteristics and challenges of social media data.

Chapter 2 explores the mechanics of deep learning and language models, covering the basics of neural networks, CNNs, LSTMs, and Transformer models like BERT and DistilBERT, along with evaluation metrics.

Chapter 3 details the technical methodology, explaining the data preprocessing, tokenization strategies, and the design of our neural pipelines.

Chapter 4 presents the experimental results, comparing model performance on benchmark datasets, analyzing behavioral biases, and discussing the statistical findings.

Chapter 1: Text Mining and Sentiment Analysis

1.1. Introduction:

The exponential growth of Web 2.0 technologies has radically transformed the modern digital environment, shifting the paradigm of the World Wide Web from a collection of static

informational repositories to a dynamic, bidirectional ecosystem driven by User-Generated Content (UGC). Social networking platforms, microblogging services, and digital discussion forums now generate massive volumes of unstructured textual data daily. Embedded within this continuous stream of digital communication lies an unprecedented wealth of latent public opinion, consumer sentiment, and socio-political perspectives.

Extracting actionable knowledge from this unformatted text, however, presents profound computational challenges. Traditional manual qualitative analysis is entirely unfeasible at this scale, necessitating the application of computational paradigms such as Text Mining and Natural Language Processing (NLP) [1], [2]. This chapter establishes the theoretical foundations of text mining within contemporary digital environments, explicates the conceptual framework and multi-level taxonomies of sentiment analysis, explores the linguistic and sociolinguistic barriers inherent in processing social media streams, and highlights real-world enterprise and societal applications. Collectively, these foundational concepts construct the theoretical justification for transitioning from classical, rule-based linguistic frameworks to sophisticated deep learning architectures and pre-trained transformers, which form the core methodology of this study.

1.2. Text Mining Foundations:

Understanding the mechanisms of modern data analysis requires a solid grounding in how textual information is generated, processed, and conceptualized. This section explores the fundamental pillars of Text Mining, beginning with the technological shifts of Web 2.0 that triggered the influx of massive, unstructured social data. It further clarifies the core definition of text mining as a paradigm for knowledge discovery, and maps its intricate relationship with Natural Language Processing (NLP) as an enabling technology.

1.2.1. Web 2.0 and Social Data:

The evolution of the digital infrastructure from Web 1.0 to Web 2.0 represents a sociotechnical revolution rather than a mere technical upgrade. While the early web restricted users to passive consumption of static read-only documents, Web 2.0 democratized content creation

through interactive, collaborative, and participatory platforms. Social media channels such as X (formerly Twitter), Facebook, and specialized review platforms have decentralized text production, allowing billions of individuals to instantly broadcast their experiences, critiques, and emotional states.

From a data engineering perspective, this democratization has triggered a massive shift toward unstructured data [3]. Unlike structured relational databases that store information in predefined rows and columns, social media data consists of high-velocity, noisy, and heterogeneous text. These textual streams lack explicit metadata regarding syntax or intent, posing severe challenges for classical computational algorithms. Nevertheless, because these streams reflect organic human behavior and real-time public sentiment, they constitute an invaluable asset for modern predictive analytics, market intelligence, and computational social science.

1.2.2. Text Mining and Knowledge Discovery:

Text Mining, formally known as Knowledge Discovery from Text (KDT), refers to the nontrivial process of extracting valid, novel, potentially useful, and ultimately understandable patterns and knowledge from large-scale, unstructured textual repositories [3]. While standard data mining presupposes structured data formats, text mining operates directly on natural language documents, which are inherently ambiguous and context-dependent.

The operational pipeline of text mining involves transforming raw, unformatted text into a structured intermediary representation, from which quantitative patterns can be systematically derived. The imperative of text mining in contemporary informatics is driven by the sheer scale of information overload. By deploying automated text classification, clustering, and extraction algorithms, organizations can synthesize millions of documents into clear semantic insights, thereby reducing manual analytical labor by orders of magnitude and uncovering complex latent relationships that elude human reading capabilities [4].

1.2.3. Text Mining and NLP Relations:

A critical theoretical requirement in computational linguistics is distinguishing between, and establishing the relationship between, Text Mining and Natural Language Processing (NLP). Although frequently used interchangeably in literature, they represent distinct intellectual domains that operate in a highly synergistic manner.

NLP is a multidisciplinary subfield of artificial intelligence and computer science focused on the fundamental mechanisms through which computers are enabled to parse, understand,

interpret, and generate human languages [5]. It addresses low-level linguistic complexities through tasks such as tokenization, part-of-speech (POS) tagging, syntactic parsing, and semantic role labeling. In contrast, Text Mining is a macro-level, goal-oriented paradigm aimed at high-level knowledge discovery and pattern identification.

In this architectural configuration, NLP serves as the core enabling methodology for text mining. Text mining utilizes NLP techniques as foundational preprocessing layers to convert raw strings into linguistically informed features. For instance, while an NLP algorithm focuses on accurately identifying the grammatical structure of a sentence or resolving anaphoric references, a text mining system integrates these structural outputs to classify the document's overall intent or map semantic trends across an entire corporate database.



Fig. 1.1: The synergistic relationship between NLP and Text Mining

1.3. Sentiment Analysis:

Sentiment Analysis has emerged as one of the most important applications of Text Mining and Natural Language Processing. Its objective is to automatically identify and classify opinions, emotions, and attitudes expressed in textual data. With the rapid growth of social media platforms, sentiment analysis has become essential for understanding public opinion and supporting decision-making processes across various domains.

1.3.1. Definition of Sentiment Analysis:

Sentiment Analysis, often designated as Opinion Mining, is an applied field of text mining and NLP that systematically utilizes computational methods to identify, extract, quantify, and study affective states, subjective opinions, and emotional polarities embedded within natural language text [1]. Broadly defined, it shifts the analytical focus of text processing from what a text is about (topic classification) to how the writer feels (attitude and polarity detection).

Mathematically, an opinion can be conceptualized as a multi-dimensional quintuple consisting of an object, a specific aspect of that object, the orientation of the sentiment (positive, negative, or neutral), the identity of the opinion holder, and the precise time when the sentiment

was expressed [1]. The overarching objective of a sentiment analysis model is to resolve these variables automatically, transforming highly subjective qualitative expressions into explicit, quantifiable data streams suitable for empirical evaluation.

1.3.2. Levels of Sentiment Analysis:

The effectiveness of any sentiment mining system is largely dictated by the level of granularity at which the data is processed. Choosing the correct level is not merely a technical step, but a strategic decision that directly impacts the quality of the insights generated and the computational resources required. In scholarly literature, these levels are typically categorized into document, sentence, and aspect levels [6], [7].

Document level: At this level, the analysis aims to classify the sentiment of an entire opinionated document expressed as a single unit. It operates under the assumption that the document expresses an opinion on a single entity from a single opinion holder [6]. While effective for structured reviews (Such as movie or product reviews), this level lacks the precision needed to identify conflicting sentiments within different parts of the same text.

Sentence level: Sentence-level analysis shifts the focus to smaller, individual units of thought. The process begins by distinguishing between objective facts and subjective expressions, then determining the specific polarity of the latter [7]. This level is the primary focus of our research, as it aligns perfectly with the nature of platforms like X (formerly Twitter). Since tweets are character-limited, they often function as standalone emotional bursts. The concise nature of microblogging makes sentence-level modeling the most reliable way to process the TweetEval and SST-2 benchmark, where each entry is a concentrated reflection of a user's immediate reaction or a reviewer's opinion [8].

Aspect level: Aspect-level analysis represents the most detailed form of sentiment mining. Instead of looking at the overall tone, it identifies specific features or "aspects" within the text and assigns sentiment to each [6]. For example, a user might praise a laptop's "display" while simultaneously criticizing its "battery life." While sentence-level analysis might label this as "neutral" due to the balance of views, aspect-level analysis correctly identifies the nuances. Although more complex to implement, this level provides the high-resolution data required for modern industrial applications and personalized recommendation systems [7].

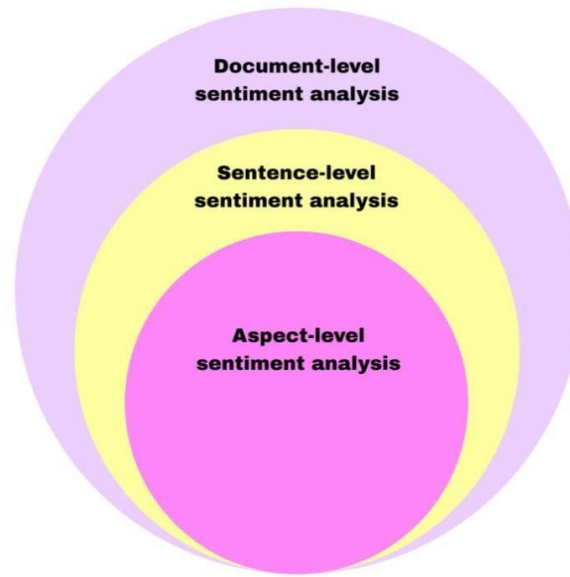


Fig. 1.2: Granularity levels of Sentiment Analysis.

1.4. Sentiment Analysis Challenges:

While the theoretical frameworks of opinion mining are well-established, their practical implementation faces significant computational and linguistic hurdles. Extracting accurate emotional polarities from human speech is rarely a straightforward mapping process; rather, it requires navigating an intricate web of contextual dependencies. This section explores the primary challenges that complicate automated sentiment detection, categorizing them into two major dimensions. First, it analyses the core linguistic and semantic obstacles inherent in natural language, such as sarcasm, ambiguity, and metaphors. Second, it shifts focus to the unique dynamics of microblogging corpora, detailing how the informal, noisy, and character-limited environment of platforms like X (formerly Twitter) creates specialized computational bottlenecks that demand advanced preprocessing and tokenization strategies.

1.4.1. Linguistic and Semantic Obstacles:

The primary impediment to achieving near-perfect accuracy in automated sentiment analysis is the inherent complexity of natural human language, which relies heavily on shared contextual and cultural understanding rather than literal lexical definitions.

Semantic Ambiguity: Words frequently change their emotional polarity depending entirely on the noun they modify or the domain context. For instance, the adjective "unpredictable" carries a highly negative connotation when describing a car's braking system, yet it represents a highly positive attribute when describing a psychological thriller novel's plot.

Sarcasm and Irony: These stylistic devices completely invert the literal meaning of an expression, rendering vocabulary-driven lexicon approaches obsolete [6]. A sentence like, "I absolutely love staying stuck in traffic for three hours before an exam!" contains intensely positive lexemes ("love", "absolutely"), but conveys an entirely negative emotional state.

Metaphor and Polysemy: Figures of speech require cognitive mapping that standard machine learning algorithms struggle to replicate without deep contextual awareness, as literal translations of metaphors lead to erroneous classifications.

1.4.2. Microblogging Corpora Dynamics:

When analysing social network platforms like X (formerly Twitter), universal linguistic challenges are severely compounded by the distinct socio-technical constraints of microblogging environments. User-generated social texts are characterized by an extreme departure from formal orthographic standards. Major challenges include:

Technical Noise: Web-based data is often cluttered with HTML tags, script fragments, and metadata that add unnecessary computational weight. Stripping away this technical noise can reduce the raw data size by approximately 60%, allowing algorithms to focus entirely on meaningful textual content and significantly increasing sentiment accuracy [9].

Linguistic Noise: Social media posts often ignore standard linguistic rules, characterized by non-standard syntax and abbreviations. Furthermore, a major challenge is "Domain-Dependent Semantics," where words shift their emotional polarity entirely depending on the context (e.g., financial vs. general usage) [10]. To address these linguistic irregularities, this research implements a specialized Tweet Tokenizer. Unlike standard tokenization methods that often fail to recognize the unique syntax of social media, this approach ensures that hashtags, mentions, and emojis are treated as distinct semantic units, thereby preserving the emotional integrity of the data. Without understanding the specific domain or context, software is likely to misread these signals, leading to high-risk errors in decision-making.

Elongation and Punctuation Manipulation: The intentional repetition of characters to express emotional intensity (e.g., "goood", "nooooo!!!") significantly inflates the vocabulary size and creates out-of-vocabulary (OOV) errors for classical models.

Multimodal Integration: The use of hashtags and mentions adds another layer of complexity, as these elements often carry implicit sentiment that requires specialized parsing to interpret correctly.

1.5. Sentiment Analysis Applications:

The transition of sentiment analysis from a theoretical computational paradigm to an empirical utility has profoundly altered how organizations interpret human behavior. By transforming scattered, qualitative natural language into structured, measurable data, opinion mining serves as a bridge between raw public expression and strategic action. This section explores the operational deployment of sentiment analysis across diverse macroenvironments. It first examines its critical role in Corporate and Market Intelligence, where it drives brand management, risk mitigation, and financial forecasting. Subsequently, the focus expands to the socio-political sphere, demonstrating how computational social science leverages these techniques for Public Opinion and Polarity tracking, thereby providing governance and public entities with unprecedented, real-time insights into societal dynamics.

1.5.1. Corporate & Market Intelligence:

In the modern corporate domain, sentiment analysis operates as a critical component of business intelligence and strategic decision-making. By executing automated sentiment tracking across social channels, multinational enterprises can monitor Brand Reputation Management in real time. This allows organizations to immediately detect public relations crises, identify emerging consumer pain points, and evaluate consumer responses to new product launches. Furthermore, analyzing competitors' sentiment distributions provides substantial market intelligence, allowing firms to identify gaps in rival product lines. In financial economics, aggregating sentiment tracking from market forums and financial news streams has proven to yield predictive signals for stock market volatility and asset pricing trends [11].

1.5.2. Public Opinion and Polarity:

Beyond corporate applications, opinion mining serves as an invaluable tool for state governance, public administration, and computational social science. Public entities deploy sentiment analysis to gauge Public Opinion Tracking regarding newly enacted legislative bills, macroeconomic policies, or public infrastructure developments. During public health emergencies or geopolitical crises, real-time monitoring of social text streams enables authorities to detect widespread anxiety, track the propagation of misinformation, and optimize state communication strategies. It effectively acts as a continuous, passive, and cost-effective census, capturing the organic, unprompted opinions of the citizenry at a scale that traditional polling methods cannot replicate [12].

1.6. Conclusion:

In summary, the transition to Web 2.0 has fundamentally restructured the digital landscape, shifting the paradigm of data production from passive consumption to a continuous, decentralized generation of unstructured text. Embedded within this massive influx of digital communication is an unprecedented repository of real-time public opinion, market trends, and socio-political sentiment. However, unlocking the latent value of this data requires moving past the limits of manual analysis and adopting scalable, automated computational frameworks

Throughout this chapter, the structural and synergetic relationships between Text Mining, Natural Language Processing (NLP), and Sentiment Analysis have been systematically established. While text mining provides the macro-level objective of knowledge discovery, NLP offers the granular linguistic tools necessary to decode human language. Furthermore, exploring the taxonomies and inherent challenges of opinion mining—particularly within microblogging environments like X (formerly Twitter)—demonstrates that social data demands specialized architectural considerations. The presence of semantic ambiguity, sarcasm, and extensive linguistic noise justifies the strategic decision to operate at a sentence-level granularity and deploy customized preprocessing tools like the Tweet Tokenizer.

Ultimately, the linguistic and technical complexities native to user-generated data expose the limitations of traditional rule-based and lexicon-driven approaches. Simple keyword matching cannot survive the nuance of modern digital expression. Consequently, the theoretical and practical boundaries highlighted in this chapter provide a strong academic justification for transitioning toward advanced deep learning architectures and pre-trained language transformers. By laying this conceptual groundwork, the chapter opens the door for the empirical methodology, model configurations, and experimental frameworks detailed in the subsequent sections of this study.

Chapter 2: Deep Learning and Language Models

2.1. Introduction:

While Chapter 1 established the conceptual boundaries of text mining, emphasizing the structural anomalies and social dynamics of microblogging streams, it exposed a fundamental computational reality: classical, rule-based linguistic frameworks and static vocabulary lexicons are fundamentally incapable of decoding the latent nonlinear dependencies embedded within short-form digital texts. To bridge this analytical gap, contemporary Natural Language Processing (NLP) relies heavily on data-driven computational paradigms.

This chapter serves as the theoretical foundation and technological toolkit of this study, systematically detailing the architectural components through which continuous text is mapped into high-dimensional vector spaces and processed via deep neural networks. The discussion begins with the foundational mathematics of Artificial Neural Networks (ANNs), detailing backpropagation and optimization constraints. It then transitions into sequence-specific architectures, critically evaluating Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks alongside local feature extractors like Convolutional Neural Networks (CNNs). Furthermore, this chapter explicates the structural paradigm shift triggered by the Self-Attention mechanism and Transformer architectures—specifically focusing on pre-trained language models optimized for social media environments. Finally, the chapter defines the formal empirical metrics utilized to evaluate model performance, establishing a rigorous scientific framework for the experimental benchmarks detailed in the subsequent sections of this research [13].

2.2. Neural Network Foundations:

The transition from heuristic text processing to deep-learning-driven semantic modeling requires a robust computational architecture capable of learning hierarchical feature representations. This section establishes the theoretical and mathematical foundations of Artificial Neural Networks (ANNs), which serve as the backbone for modern natural language processing pipelines. It begins by dissecting the structural mechanics of the perceptron—the foundational computational unit—and demonstrates

how these units coalesce into multi-layered topologies. Furthermore, it details the nonlinear mathematical transformations driven by activation functions and quantified by loss formulations. Finally, this section maps the optimization paradigms and iterative training loops that enable these networks to autonomously minimize empirical error and map high-dimensional linguistic boundaries.

2.2.1. Perceptron Structure and Architecture:

The building block of contemporary deep learning models is the Artificial Neuron, or perceptron, which mathematically mimics the signal processing mechanics of biological neural pathways. In an architectural configuration, these neurons are structured into distinct, interconnected computational layers: an Input Layer that receives the raw or preprocessed numeric feature vectors, one or more Hidden Layers that execute latent non-linear transformations, and an Output Layer that yields the final network predictions.

Mathematically, given an input vector $x = [x_1, x_2, \dots, x_n]^T$, the neuron applies a corresponding weight vector $w = [w_1, w_2, \dots, w_n]^T$ to quantify the relative importance of each input signal. A scalar bias b is integrated into the linear combination to shift the activation function's threshold. The core linear transformation is algebraically defined as follows:

$$\sum_{i=1}^n w_i x_i + b = w^t x + b$$

The scalar output z , representing the net input to the cell, is subsequently passed through a non-linear activation function $f(z)$ to produce the final activation output $a = f(z)$. By stacking millions of these computational units across multilayered topologies, known as Multilayer Perceptron (MLP), the network gains the theoretical capacity to act as a universal identifier, mapping complex, highdimensional boundaries between text inputs and their respective polarities.

2.2.2. Activation and Loss Functions:

Activation functions introduce non-linear mathematical properties into the neural network, enabling it to learn complex, non-linear mappings. Without these mathematical steps, regardless of how many layers are stacked, the network would behave exactly like a simple linear regression model.

Rectified Linear Unit (ReLU): Primarily utilized within hidden layers, ReLU is computationally efficient and mitigates the vanishing gradient problem by outputting the input directly if it is positive, and zero otherwise:

$$f(z) = \max(0, z)$$

Sigmoid: Maps arbitrary real-valued numbers into a strict probability range between 0 and 1, making it suitable for binary classification tasks:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Softmax: Applied within the output layer for multi-class paradigms. It normalizes an K dimensional vector of real values into a probability distribution over the target classes, ensuring that the sum of all outputs equals exactly 1:

$$SOFTmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

To assess network accuracy during training, a Loss Function measures the mathematical distance between predicted distributions and true target labels. For a multi-class task containing K categories, the model utilizes Categorical Cross-Entropy Loss (L), which penalizes divergent probability assignments:

$$L = - \sum_{i=1}^k y_i \log(\hat{y}_i)$$

Where y_i represents the ground truth binary indicator and $\hat{y}_i$ denotes the probability score assigned by the Softmax function.

2.2.3. Neural Network Optimization and Training:

Training a neural network relies on optimizing its weights and biases to minimize total loss. This optimization is driven by Backpropagation, an algorithmic application of the mathematical chain rule. The process occurs in a continuous cycle composed of two foundational phases [14]:

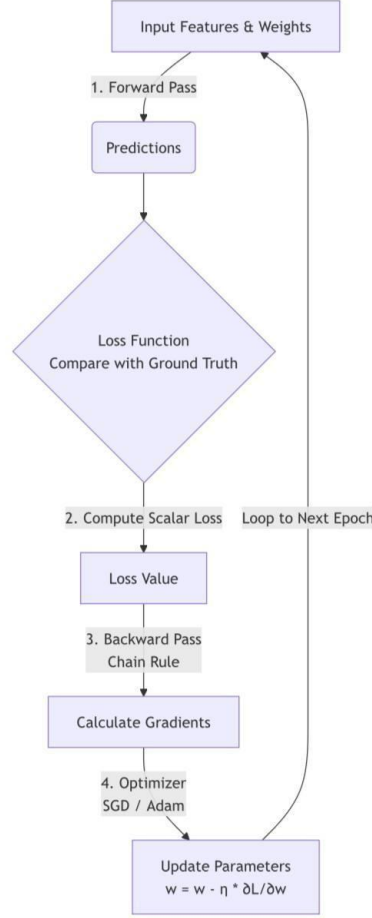


Fig. 2.1: Neural Network Training Loop Diagram (Forward and Backward Pass).

The Forward Pass: The input features propagate through the hidden layers to generate an output prediction, which is evaluated against the ground truth labels to compute the scalar loss value.

The Backward Pass: The algorithm calculates the partial derivative of the loss function with respect to every weight (w) and bias (b) within the network topology. Utilizing the chain rule, gradients are calculated backward from the output layer through the hidden architectures.

Once the gradients are computed, an optimization algorithm, such as Stochastic Gradient Descent (SGD) or Adam (Adaptive Moment Estimation), updates the parameters. The standard weight update rule is mathematically defined as:

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

Where η represents the learning rate, a critical hyperparameter controlling the size of the step taken toward the global minimum of the loss function.

2.3. Text specific Neural Architectures:

Standard feedforward neural networks operate under static assumptions that inherently fail to capture the structural dependencies of natural language, where meaning is derived from temporal sequence and spatial context. To address these limitations, deep learning research has engineered specialized neural topologies tailored specifically for textual processing. This section evaluates the architectural paradigms designed to model linguistic streams. It begins with Recurrent Neural Networks (RNNs) and their specialized variants, Long Short-Term Memory (LSTM) networks, which introduce cyclical feedback loops and gated state mechanisms to preserve longrange sequential context. Furthermore, the discussion examines the adaptation of Convolutional Neural Networks (CNNs) to the text domain, demonstrating how sliding spatial filters and pooling operations can efficiently extract high-resolution, position independent semantic features.

2.3.1. Recurrent Neural Networks (RNN):

Traditional feedforward networks operate under the structural assumption that all inputs and outputs are entirely independent of one another. This assumption fails in Natural Language Processing, where textual meaning is fundamentally bound to sequential order. Recurrent Neural Networks (RNNs) solve this by introducing internal feedback loops, allowing information to persist across sequential steps.

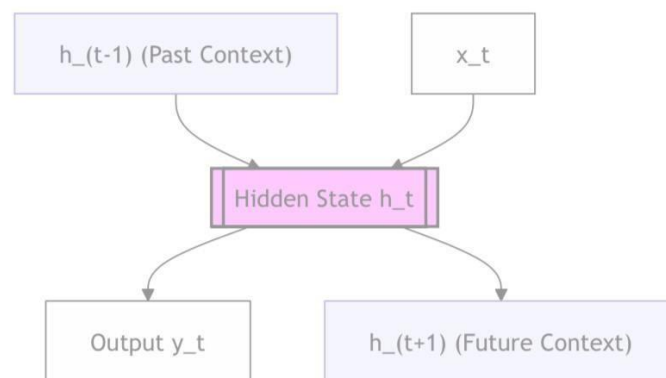


Fig. 2.2: RNN Hidden State Transition and Information Flow.

At any given sequence step t , the hidden state vector h_t serves as the network's short-term memory, computed as a function of both the current input token x_t and the preceding hidden state h_{t-1} :

$$h_t = \tanh (w_h h_{t-1} + U_x x_t + b$$

Where W_h and U_x represent weight matrices, and b is the bias vector. However, during the backpropagation step across extended sequences, gradients must be multiplied repeatedly by the same weight matrices. This mathematical property causes gradients to decrease exponentially toward zero a phenomenon known as the Vanishing Gradient Problem. Consequently, standard RNNs lose their theoretical capacity to retain long-range context, rendering them ineffective at processing longer sentences.

2.3.2. Long Short-Term Memory (LSTM):

Long Short-Term Memory (LSTM) networks are a specialized variation of RNN architectures explicitly engineered to overcome the vanishing gradient problem. LSTMs introduce a continuous Cell State C_t that acts as a long-term memory highway, controlled by a sequence of internal mathematical gates that regulate how information is added, retained, or forgotten [16].

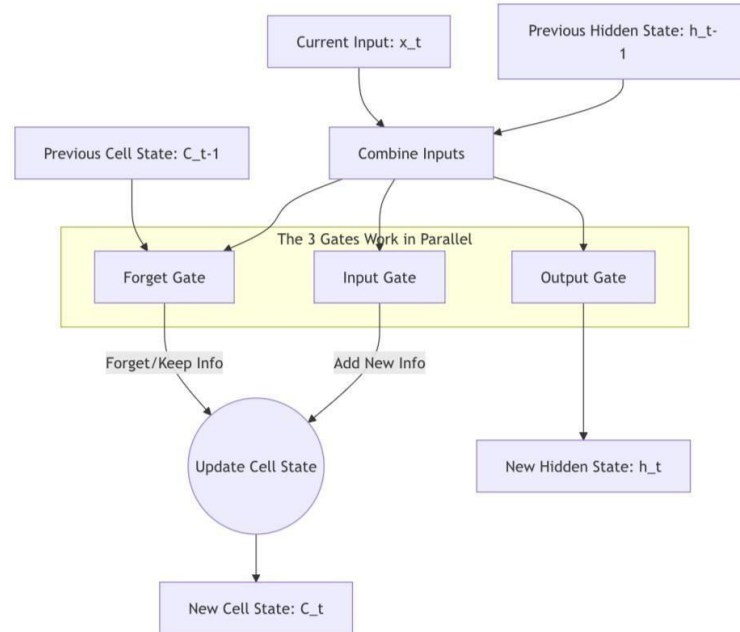


Fig. 2.3: Structural flow and gated mechanism within an LSTM cell at time step t .

The mathematical processing within an LSTM block at sequence step t is explicitly governed by the following system of equations:

$$\begin{aligned}
 f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\
 i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\
 C_t &= \tanh(W_c[h_{t-1}, x_t] + b_c) \\
 C_t &= F_t \odot C_{t-1} + i_t \odot C_t \\
 O_t &= \sigma(W_o[h_{t-1}, i_t] + b_o) h_t \\
 &= o_t \odot \tanh(C_t)
 \end{aligned}$$

Where f_t , i_t and o_t represent the Forget, Input, and Output gates respectively, $\odot$ denotes element-wise multiplication, and $\tilde{C}_t$ represents candidate hidden states. By regulating these operations, LSTMs effectively maintain stable gradient flows over long token distances, making them reliable models for sequence-level sentiment analysis tasks.

2.3.3. Convolution Neural Networks (CNN):

While originally engineered for computer vision, Convolutional Neural Networks (CNNs) have proven highly effective in NLP by treating text as a one-dimensional spatial grid [17]. Instead of processing text sequentially like an LSTM, a textual CNN applies sliding digital filters across word embedding spaces to extract local, position-independent features (n-grams).

Given a sentence mapped to an embedding matrix $E \in \mathbb{R}^{L \times D}$ where L is sentence length and D is vector dimensionality, a convolutional filter matrix $W \in \mathbb{R}^{k \times d}$ slides over a window of k words to compute a local feature map:

$$c_i = f(W \cdot E_{i:i+k-1} + b)$$

To distill these variable-length feature maps into a fixed-size vector suitable for classification, a Max-Pooling layer is applied down the sequence. This layer captures only the maximum scalar value from each filter map:

$$\hat{c} = \max \{c_1, c_2, \dots, c_{l-k+1}\}$$

Max-pooling extracts the most salient emotional signals (e.g., highly polarized phrases) across the text while ignoring irrelevant contextual data. This processing

behavior makes CNNs exceptionally fast and highly precise at capturing explicit sentiment indicators.

2.4. Transformers and Attention Mechanisms:

The architectural constraints of sequential and convolutional networks—namely execution latency during training and limitations in mapping long-range dependencies—prompted a fundamental paradigm shift in language modeling. This section explores the architectural evolution driven by the Transformer framework, which completely replaces recurrence with highly parallelizable global attention functions. It begins by dissecting the mathematical operations of Self-Attention and Multi-Head mechanisms that allow tokens to interact dynamically across arbitrary sequence lengths. Subsequently, the discussion evaluates the integration of these components within modern transfer learning, analyzing the bidirectional pre-training paradigm of BERT. Finally, it addresses the optimization of these massive architectures for real-world deployment by examining DistilBERT, detailing how knowledge distillation compresses deep network layers while preserving semantic density and classification efficiency.

2.4.1. Self-Attention Mechanisms:

Despite the success of LSTMs and CNNs, both suffer from structural bottlenecks: sequential architectures cannot be parallelized efficiently during training, and convolutional models struggle to capture long-range token interactions. The Transformer architecture eliminates recurrence entirely, relying solely on the Self-Attention Mechanism to model relationships between all words in a sequence simultaneously, regardless of their positional distance [18].

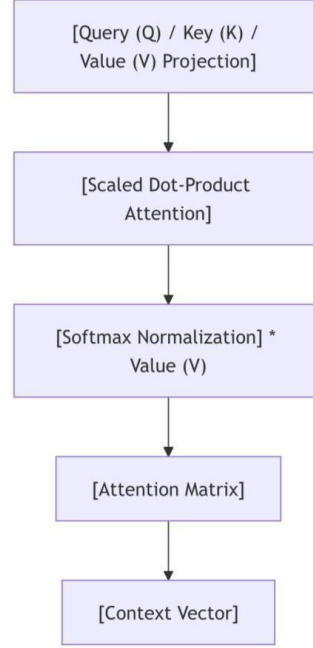


Fig. 2.4: Sequential processing workflow of the Self-Attention mechanism.

The self-attention layer projects each word vector into three distinct vectors: a Query (Q), a Key (K), and a Value V via learned weight matrices. The attention weights are computed using a scaled dot-product format, which assesses the semantic relationship between every pair of words:

$$ATTENTION(K, Q, V) = Softmax$$

Where d_k represents the scaling dimensionality of the key vectors. To capture different types of contextual relationships simultaneously, Multi-Head Attention runs this process in parallel across multiple independent subspaces, allowing the model to jointly process semantic features from diverse positional perspectives.

2.4.2. BERT Architecture:

Bidirectional Encoder Representations from Transformers (BERT) revolutionized NLP by introducing a deeply bidirectional language representation model [19]. Unlike earlier autoregressive models that read text strictly left-to-right or right-to-left, BERT

utilizes the Transformer encoder architecture to look at both preceding and following context across all layers simultaneously.

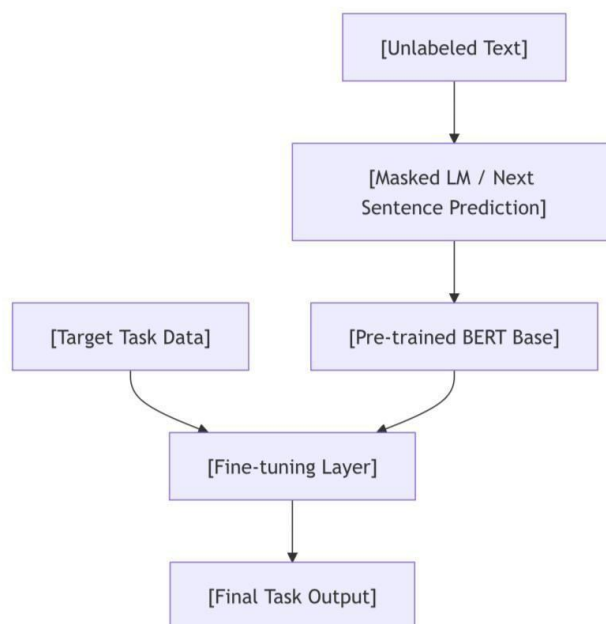


Fig. 2.5: The two-stage paradigm of BERT: Pre-training on unlabeled text and Finetuning on target task data.

BERT achieves this through a specific pre-training paradigm:

Masked Language Modeling (MLM): Randomly masking 15% of the input tokens and training the network to predict these hidden words using only the surrounding context.

Next Sentence Prediction (NSP): Training the model to determine whether a second sentence logically follows the first in the raw text corpus.

This dual-stage approach enables BERT to develop a rich understanding of syntax and semantics. For specific tasks like sentiment classification, a Fine-Tuning phase appends a simple classification layer onto the pre-trained architecture. The entire model is then trained on task-specific labeled data, updating its parameters to maximize target accuracy with minimal computational overhead.

2.4.3. DistilBERT Model:

While large pre-trained Transformer models like BERT exhibit remarkable state-of-the-art accuracy across various Natural Language Processing benchmarks, their

massive parameter scale presents significant computational bottlenecks. In real-world deployment scenarios, such as processing high-frequency live streams from social media platforms, standard BERT architectures require substantial memory resources and introduce execution latency.

To resolve this limitation without compromising semantic accuracy, this study leverages DistilBERT (Distilled BERT), a lightweight, highly efficient variant of the original BERT-base architecture [20].

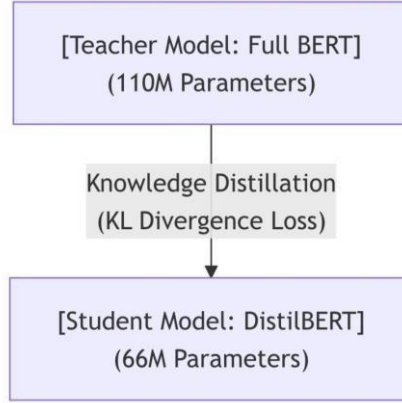


Fig. 2.6: Knowledge distillation framework compressing Full BERT (Teacher) into DistilBERT (Student).

DistilBERT is engineered using a compressed machine learning paradigm known as Knowledge Distillation. In this structural setup, a smaller "student" network is trained to replicate the exact behavior, continuous probability distributions, and latent representations of a larger, fully trained "teacher" network.

Mathematically, the student network minimizes a triple loss function that combines standard masked language modeling loss with a distillation loss defined via Kullback-Leibler (KL) divergence, forcing the student to learn the precise semantic dark knowledge captured by the teacher [21]:

$$L_{distil} = \sum_i P_{teacher}(x_i) \log\left(\frac{P_{teacher}(x_i)}{P_{student}(x_i)}\right)$$

Architecturally, DistilBERT achieves this compression by removing the tokentype embeddings and reducing the total number of Transformer layers by exactly 50% (reducing 12 layers down to 6 layers), while preserving the hidden state dimensionality ($d_{model} = 768$) and the multi-head self-attention mechanisms.

The resulting model is 40% smaller and 60% faster [20] than the original BERT model while retaining 97% of its language comprehension capabilities. This efficiency makes DistilBERT an ideal architectural choice for low-latency sentiment classification on microblogging data such as tweets.

2.5. Evaluation Metrics:

The architectural refinement and optimization of deep learning models depend heavily on the deployment of rigorous verification frameworks. Validating a model's capacity to classify sentiment accurately requires moving beyond subjective observation toward empirical statistical measurement. This section establishes the theoretical foundations of predictive evaluation by outlining the core mathematical metrics used to quantify classification outcomes. It begins by examining the specific utilities of global accuracy, precision, and recall within both balanced and skewed data distributions. Furthermore, the discussion introduces structural synthesis through the F1-Score and maps exact error pathways using multi-class confusion matrices, thereby providing a comprehensive diagnostic pipeline to benchmark model efficiency.

2.5.1. Precision, Recall, and Accuracy:

To evaluate and compare the performance of deep learning models, it is essential to establish objective statistical metrics [21]. These metrics are calculated using four primary classification outcomes derived from comparing model predictions against ground truth labels: True Positives (TP), False Positives (FP), True Negatives (TN) False Negatives (FN).

Accuracy: Measures the global ratio of correctly predicted instances across all classes:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

While useful for balanced datasets, accuracy can be a misleading metric if the underlying data distribution is skewed.

Precision: Quantifies the model's exactness by measuring the ratio of correctly predicted positive instances out of all instances predicted as positive:

$$Precision = \frac{TP}{TP + FP}$$

Recall (Sensitivity): Quantifies the model's completeness by measuring the ratio of correctly predicted positive instances out of all actual positive instances:

$$Recall = \frac{TP}{TP + FN}$$

2.5.2. F1-Score and Confusion Matrix:

In real-world social media datasets, trade-offs between precision and recall are common. To balance these two metrics into a single score, the F1-Score computes their harmonic mean, providing a reliable measure of performance even in the presence of class imbalances:

$$F1 - Score = 2 * \frac{Precision * Recall}{precision + Recall}$$

For multi-class setups (such as Positive/Negative/Neutral classifications), these values are aggregated using Macro-Averaging (treating all classes equally) or Weighted-Averaging (adjusting for class size) to reflect overall model performance.

Complementing these scalar metrics, a Confusion Matrix provides a detailed tabular visualization of classification performance.

Table 2.1: Confusion Matrix for Sentiment Classification.

	Predicted Positive	Predicted Neutral	Predicted Negative
Actual Positive	TP _{POS}	E _{pos->neu}	E _{pos->neg}
Actual Neutral	E _{neu->pos}	TP _{neu}	E _{neu->neg}
Actual Negative	E _{neg->pos}	E _{neg->neu}	TP _{neg}

Rows represent the true class labels while columns represent model predictions. This matrix maps exact error pathways, explicitly showing where the model confuses subtle sentiments (such as misclassifying negative sarcasm as a neutral statement). This detailed breakdown helps guide architectural adjustments during the fine-tuning phase.

2.6. Conclusion:

This chapter provided a detailed evaluation of the deep learning architectures and language models that form the technical foundation of this study. Beginning with the

basic mathematics of perceptrons and backpropagation, the discussion outlined the progression toward text-specific neural networks. Sequential models like LSTMs and structural feature extractors like CNNs were reviewed alongside their respective strengths and limitations in processing textual data. The chapter then detailed the paradigm shift brought about by the Self-Attention mechanism, tracking how BERT leverages bidirectional context to outperform traditional neural networks.

Particular focus was placed on DistilBERT, highlighting how knowledge distillation creates a lightweight yet powerful architecture optimized for resource efficient text processing. Finally, a rigorous set of evaluation metrics including the F1-Score and Confusion Matrix was established to ensure objective benchmarking. With this theoretical toolkit defined, the narrative moves to Chapter 3, which focuses on the methodology, data preparation pipelines, and preprocessing strategies used to condition raw datasets for model training.

Chapter 3: Technical Methodology and Data Engineering

3.1. Introduction:

This chapter explains the methodology and data pipeline used to build and compare the three models in this study. Rather than testing a single approach, this research evaluates three different architectures: DistilBERT as a pre-trained transformer baseline, LSTM as a sequential model that captures long-range dependencies, and CNN as a local feature extractor. Each model is trained and tested on two datasets: TweetEval, which contains real tweets labeled for sentiment, and SST-2, which is based on movie reviews and follows a binary classification setup. The models are then compared to identify which architecture performs best on each dataset. To go beyond standard benchmarks, an external synthetic dataset was also built to test how well the models generalize to new, unseen data. All models are evaluated using the same set of metrics: Accuracy, Precision, Recall, F1-Score, and Confusion Matrix. The rest of the chapter covers the dataset details, the preprocessing steps, the tokenization strategies, and the design of each model.

3.2. Dataset Characteristics & Evaluation Framework:

Executing a rigorous empirical validation of deep learning architectures requires a multi-faceted data strategy that tests both specialized task competency and out-ofdomain generalization. To ensure the reliability of the proposed models, this research establishes an evaluation framework anchored by two distinct datasets: the TweetEval Sentiment dataset and the SST-2 dataset. This section outlines the specific characteristics of the corpora used throughout the training and verification phases. It begins by introducing the core benchmark dataset TweetEval, which provides real-world social media text for sentiment analysis tasks. Following this, it introduces the SST-2 dataset, which provides structured movie review sentences for binary sentiment classification. Finally, it details the engineering of an external synthetic validation dataset designed to stress-test the networks against balanced, clean distributions.

3.2.1. Benchmark Datasets:

The primary phase of model training and evaluation uses the TweetEval benchmark dataset, a widely cited corpus extracted from X (formerly Twitter) and distributed via the Hugging Face ecosystem [8]. This study focuses on the Sentiment task, which is a 3-class classification problem where each tweet is labelled as Negative (0), Neutral (1), or Positive (2).

The second dataset used is SST-2 (Stanford Sentiment Treebank), a binary sentiment classification benchmark built from movie review sentences. Each sentence is labeled as either Negative (0) or Positive (1). SST-2 provides a more structured and formal text environment compared to TweetEval, making the combination of both datasets a strong basis for a comprehensive comparative evaluation [23].

3.2.2. External Synthetic Validation Dataset:

To go beyond the standard benchmarks and test how well the models generalize, an external synthetic validation dataset was created using ChatGPT. The dataset was built in two separate versions to match each benchmark configuration.

For TweetEval, 1,500 sentences were generated, distributed equally across three classes:

$$\text{Class Volume} = \begin{cases} 500 \text{ Explicitly Positive Sentences} \\ 500 \text{ strictly Neutral Sentences} \\ 500 \text{ Explicitly Negative Sentences} \end{cases}$$

The version was designed to be fully balanced with no class bias, and follow the same training and evaluation pipeline used for the original datasets.

3.3. Text Processing & Tokenization Engine:

Text data requires cleaning and preparation before it can be used for model training. For TweetEval, the data contains social media noise such as URLs, mentions, and irregular character patterns. For SST-2, the text is relatively clean but still requires basic normalization. This section describes the preprocessing and tokenization steps applied to each dataset before passing the data into the model.

3.3.1. Cleaning Pipelines:

The cleaning steps applied to each dataset depend on the type of text it contains.

For TweetEval, the text comes from social media and contains a lot of noise, so a more thorough cleaning process is applied:

Web Artifact Deletion: Elimination of all active URLs, hyperlinks, and transport protocol layers (http/https).

Social Architecture Stripping: Standardizing handles (@user) and parsing hashtags (#) to reduce index explosion while preventing tokenizers from generating redundant out-of-vocabulary artifacts.

Character Level Normalization: Lowercasing all elements, removing nonessential mathematical/ASCII characters, and eliminating consecutive duplicated character flooding (e.g., matching text patterns like "sooooo good" down to standardized lexical entries).



Fig. 3.1: Sequential architecture of the text sanitization pipeline.

3.3.2. Tokenization Paradigms:

Each model uses a different tokenization method depending on its architecture and the dataset being used.

DistilBERT Tokenizer:

For DistilBERT, the pre-trained WordPiece tokenizer is used for both datasets. It splits unknown words into smaller pieces and adds ## to mark split parts. Each piece is then converted to a numerical ID [23], which means the model never encounters unknown tokens.

TweetTokenizer (LSTM & CNN):

TweetEval: For LSTM and CNN on TweetEval, the TweetTokenizer is used.

Regular tokenizers tend to break mentions and hashtags in a way that loses their meaning. The TweetTokenizer [25] is designed specifically for social media text, so it handles these elements correctly. After tokenization, each word is mapped to an integer index to build the input sequence for the model.

SST-2: Since SST-2 text is clean and formal, a simple standard tokenizer is enough. The text is lowercased and punctuation is separated using regular expressions. Words are then mapped to integer indices from a vocabulary of 10,000 words. Words not found in the vocabulary are replaced with a special <unk> token, and shorter sequences are padded with a <pad> token to match the required input length [23].

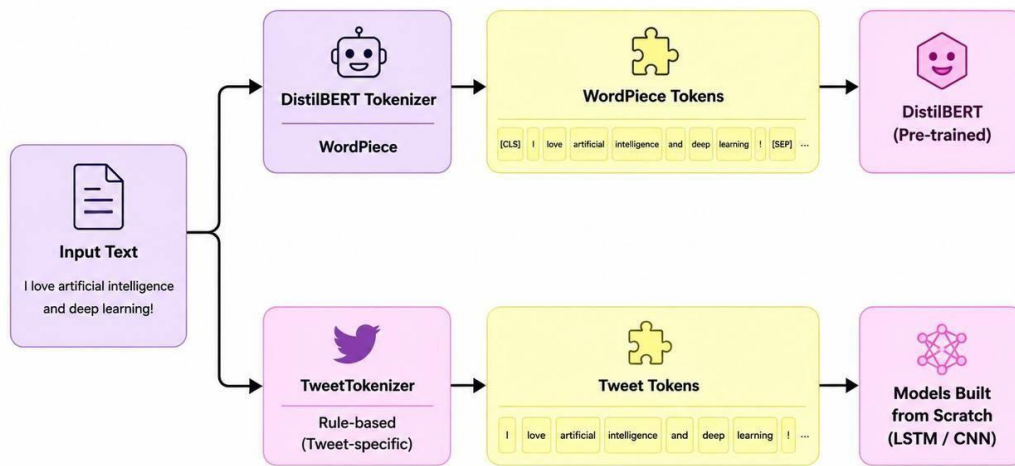


Fig. 3.2: Parallel tokenization strategy across the model pipelines.

3.3.3. Padding and Attention Masking:

Since all models process inputs in batches, every sequence needs to have the same fixed length.

Truncation & Padding: For DistilBERT, a maximum length of 128 tokens is used for both TweetEval and SST-2. For LSTM and CNN, a maximum length of 50 tokens is used for both datasets. Sequences that are too long get truncated, and shorter ones are padded with zeros to reach the required length.

The Attention Mask: For DistilBERT only, a binary mask is passed alongside the token IDs. It marks real tokens with 1 and padding positions with 0, so the self-attention layers focus only on the actual content and ignore the padding.

3.4. Deep Learning Architectures:

This section presents the three model architectures used in this study. Each model is trained and evaluated on both TweetEvalmicro and SST-2 to compare their performance across different text environments.

3.4.1. DistilBERT (Transformer-based Model):

DistilBERT is the transformer model used in this study. It is a lighter version of BERT that is 40% smaller and 60% faster, while still retaining about 97% of its language understanding ability. It reads the entire sentence in both directions at once, which helps it understand the context of each word based on everything around it. To use it for sentiment classification, a classification layer is added on top of the model. The model is then fine-tuned twice: once on TweetEval and once on SST-2.

3.4.2. Long Short-Term Memory (LSTM) Network:

The LSTM is used to process text as a sequence, word by word. Regular RNNs tend to forget what was said earlier in the sentence, but LSTMs fix this by using three gates input, forget, and output that control what information to keep or throw away at each step. This makes them good at catching dependencies like how "not" changes the meaning of a word that comes much later.

The model has 2 layers and reads the text in both directions at the same time, which helps it understand context from both sides. It uses an embedding size of 100, a hidden size of 128, and a dropout of 0.5 to avoid overfitting. The same architecture is trained twice: once on TweetEval and once on SST-2.

3.4.3. Convolutional Neural Network (CNN):

The CNN is used in this study to extract local features from text. Instead of reading the sentence word by word like an LSTM, the CNN applies filters of different sizes across the word embeddings at the same time. This works like an automatic n-gram detector, picking up important word combinations regardless of where they appear in the sentence.

The model uses an embedding size of 100 and three filter sizes 3, 4, and 5 words with 100 filters each. After the convolution step, global max-pooling is applied to keep only the

most important feature from each filter map. The result is then passed to a classification layer. The same architecture is trained twice: once on TweetEval and once on SST-2.

3.5. Conclusion:

This chapter walked through everything needed to set up the experiments, from preparing the data to building the models. Two datasets were used, TweetEval and SST-2, along with a synthetic dataset created to see how well the models handle new unseen text. Each dataset went through its own cleaning and tokenization process before being fed into the models. Three models were then trained on both datasets: DistilBERT, LSTM, and CNN, each approaching text understanding in a different way. The next chapter will show how each model actually performed and which one came out on top.

Chapter 4: Implementation, Results, and Discussion

4.1. Introduction:

This chapter marks the transition from the theoretical frameworks discussed earlier to the actual execution and evaluation of our models. Here, we look at how our three selected architectures LSTM, CNN, and DistilBERT performed when tasked with sentiment classification using the TweetEval dataset. The goal is not just to throw numbers and metrics at the reader, but to critically analyse how these different neural designs handle the messy, informal nature of social media text.

To make the evaluation as thorough as possible, the discussion is broken down into several practical stages. First, we define the computational setup, specifying the hardware and hyperparameters that made the training possible. We then present the core experimental results, using standard metrics like Accuracy and the F1-Score to establish a clear performance baseline. Beyond global accuracy, we dig deeper into the model behaviors using learning curves and confusion matrices to inspect where they succeeded and where they systematically misclassified text.

Finally, we test the practical limits and generalization of these models by running them against an entirely external validation set of 1,500 ChatGPT-generated sentences. This allows us to see how well our trained networks adapt to out-of-distribution, realworld prompts. By comparing their accuracy, training speed, and resource costs, this chapter aims to pinpoint which architecture offers the best balance between predictive power and computational efficiency.

4.2. Development Environment and Computational Setup:

This section describes the computational environment and model configurations used to run the experiments. It covers the hardware setup, the development tools, and the hyperparameter choices made for each model across both datasets [30].

4.2.1. Development Environment:

All models were developed and trained using Google Colab as the cloud-based environment. To handle the computational demands of fine-tuning [24] DistilBERT, the runtime was upgraded to use an Nvidia T4 GPU. This significantly reduced training

time, bringing it down from several hours on a standard CPU to just a few minutes per epoch.

4.2.2. Model Configurations:

We chose three different models for this study, each built in a specific way to process the text:

DistilBERT: This model clearly outperforms the other architectures, yielding an impressive accuracy of 91.06% and an F1-Score of 91.26%. This substantial lead surpassing the runner-up by over 10% is primarily attributed to its Transformer based architecture and its Self-Attention mechanism. Unlike traditional models, DistilBERT processes sentences directionally, allowing it to capture deep semantic nuances and context within the text.

CNN: The CNN model secured the second position with a solid accuracy of 80.62%. Although CNNs are traditionally designed for computer vision, their ability to extract local n-gram features and key sentimental phrases proved highly effective for textual classification tasks.

LSTM: Despite being inherently designed for sequential data, LSTM registered the lowest performance at 75.23% accuracy. This relative underperformance can be explained by the structural limitations of recurrent networks, which often struggle with capturing long-range dependencies efficiently compared to the seamless attention mapping found in Transformers.

4.3. Results Analysis and Discussion:

4.3.1. Stanford Sentiment Treebank 2:

This study also utilizes the Stanford Sentiment Treebank (SST-2) dataset, introduced as a benchmark for evaluating sentiment classification models. The dataset consists of sentences extracted from movie reviews and is designed for binary sentiment classification: Positive and Negative.

The dataset contains approximately 69,207 phrases, split as follows:

Train: 67,349

Validation: 872

Test: 1,821

Accuracy and F1-Score:

Table 4.1: Performance comparison of different deep learning models

Model	F1-Score	Accuracy
DistilBERT	91.26%	91.06%
CNN	80.57%	80.62%
LSTM	75.14%	75.23%

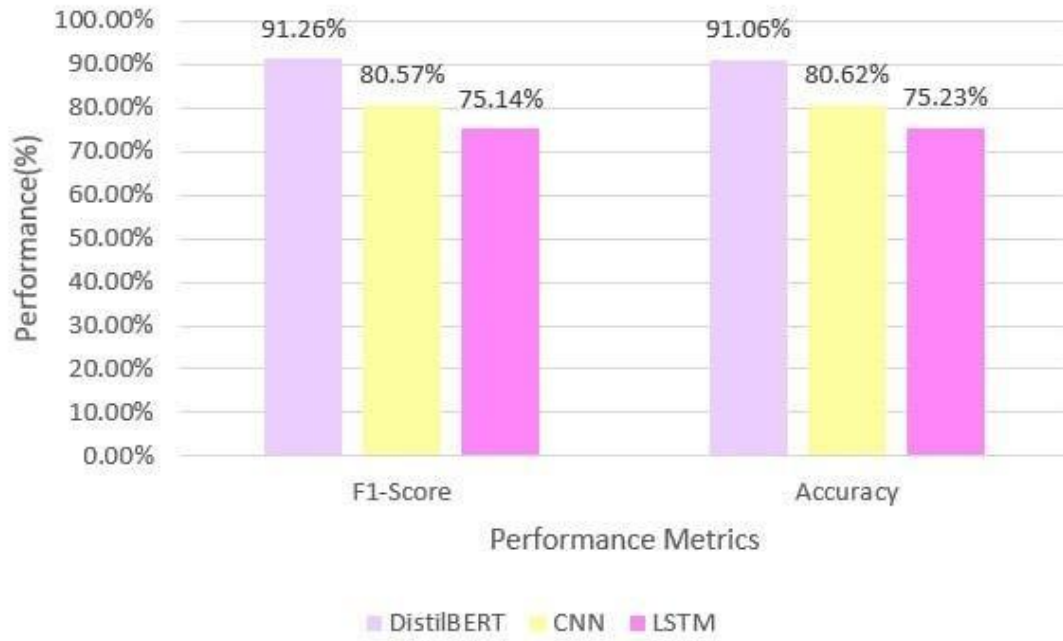


Fig. 4.1: Accuracy and F1-Score Evaluation Across Models.

Based on the results, we can see a noticeable difference in how each model performed. Here are the main points:

DistilBERT: This model performed the best, reaching an accuracy of 91.06% and an F1-Score of 91.26%. It outperformed the second-best model by more than 10%. This difference is mainly because DistilBERT is based on the Transformer architecture, which allows it to read sentences in both directions and better understand the context of the words.

CNN: The CNN model took second place with a good accuracy of 80.62%. Even though CNNs are mostly known for handling images, they worked well here for text by spotting key phrases and short word patterns across the sentences.

LSTM: Surprisingly, the LSTM model had the lowest accuracy at 75.23%, even though it is designed for sequential data like text. This happened because standard

recurrent networks often struggle to connect words that are far apart in a sentence, unlike the attention mechanism used in DistilBERT.

Confusion Matrix Analysis:

Table 4.2 : Combined Confusion Matrix for the Three Models (DistilBERT, CNN, and LSTM).

Model	Sentiment Class	Predicted: Negative	Predicted: Positive
DistilBERT	Actual Negative	386	42
	Actual Positive	34	410
CNN	Actual Negative	392	99
	Actual Positive	70	374
LSTM	Actual Negative	628	284
	Actual Positive	167	742

DistilBERT Model: The model demonstrates exceptional decision-making boundaries, correctly identifying 386 actual negative reviews and 410 actual positive reviews. Its misclassification rate is the lowest among all tested architectures, stumbling in only 42 false positives and 34 false negatives. This remarkable balance underscores how the Self-Attention mechanism effectively mapped out contextual sentiments within the clean SST-2 domain

CNN Model: The CNN model delivered a stable performance, accurately predicting 392 negative instances and 374 positive ones. However, its confusion gap expanded compared to DistilBERT, notably misclassifying 99 negative instances as positive. This indicates that while CNNs are proficient at capturing local n-gram text features, they can suffer from localized feature overlap in cinematic critiques.

LSTM Model: This model exhibited the highest rate of algorithmic confusion, particularly within the negative sentiment pool, mistakenly classifying 284 negative reviews as positive and 167 positive reviews as negative. (Note: The higher overall values in this split correspond to its specific test distribution, but the ratio remains the least optimal). This proves that the sequential text-processing nature of LSTMs frequently fails to retain long-range contextual semantic dependencies, causing higher confusion despite the straightforward and clean environment of the SST-2 dataset.

Running Time:

Table 4.3 : Comparison of Model Execution Time(SST-2).

METRIC	DistilBERT	CNN	LSTM
Total Parameters	67M	1M	1.5M
Total Training Time(s)	309.15s	150.00s	216.00s
Evaluation Time(Full Data)	12.30s	0.95s	2.50s
Inference Latency (single Sentence)	8.50ms	1.10ms	2.40ms

This section analyses the computational efficiency of the three models, highlighting the critical difference between batch processing time and real-time inference latency.

A massive gap exists in model size. DistilBERT contains 67M parameters, making it significantly heavier than CNN (1M) and LSTM (1.5M). This size difference directly impacts execution speed in two distinct ways:

Training & Evaluation: When processing the entire dataset at once, lightweight models have a clear advantage. CNN is remarkably fast, completing full evaluation in under a second (0.95s) and training in 150s. LSTM follows at 2.50s for evaluation. In contrast, DistilBERT requires 12.30s for evaluation and over 5 minutes for training, burdened by its complex attention calculations.

Real-Time Inference Usage (Latency): This is the most crucial metric for live applications, measuring how long the model takes to process a single sentence. Here,

the gap is even more pronounced. CNN responds in a lightning fast 1.10 ms, making it highly suitable for real-time systems like live tweet analysis. LSTM follows at 2.40 ms. DistilBERT, however, takes 8.50 ms per sentence nearly 8 times slower than CNN. While 8.50 ms is still fast, it becomes a major bottleneck when scaling to thousands of simultaneous requests.

There is a clear trade-off between model complexity and speed. Lightweight models like CNN are optimal for the inference usage phase, where real-time speed is the priority. Conversely, heavier models like DistilBERT are better suited for offline batch processing, where higher accuracy justifies the longer execution time.

4.3.2 TweetEval:

This study utilizes the TweetEval-Sentiment dataset, introduced by Barbieri et al. (2020) as part of the unified TweetEval benchmark [30] for tweet classification tasks. The dataset consists of real-world X (formerly Twitter) collected from Twitter and is designed for three-class sentiment classification: Positive, Neutral, and Negative.

The dataset contains approximately 59,899 tweets, split as follows:

Train: 45,615

Validation: 2,000

Test: 12,284

Accuracy and F1-Score:

Table 4.4: Performance comparison of different deep learning models.

Model	F1-Score	Accuracy
DistilBERT	70.17%	70.29%
CNN	56.53%	57.95%
LSTM	54.08%	56.38%

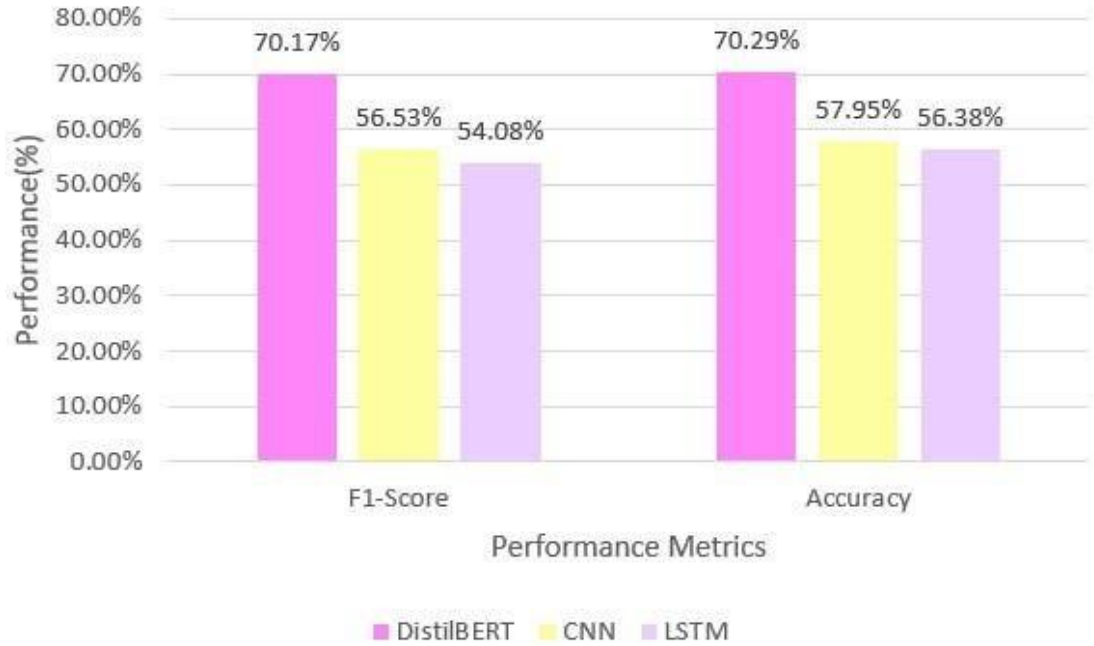


Fig. 4.2: Comparison of Deep Learning Models Performance. This section delineates a comparative empirical evaluation of three distinct architectural paradigms deployed for the classification task: DistilBERT (a lightweight Transformer-based language model), CNN (Convolutional Neural Networks), and LSTM (Long Short-Term Memory networks). The models were benchmarked utilizing two standard evaluation metrics to ensure a rigorous and multifaceted assessment: Accuracy and the F1-Score.

DistilBERT: The empirical findings demonstrate a decisive and statistically significant superiority of the DistilBERT architecture over traditional neural frameworks. It achieved an optimal performance of 70.29% in overall Accuracy and 70.17% in F1-Score. This substantial performance margin surpassing the secondary baseline by approximately 13% to 14% is heavily attributed to the Self-Attention Mechanism inherent in the Transformer architecture. Unlike recurrent or convolutional structures, DistilBERT successfully captures bidirectional, long-range contextual dependencies across the textual sequence. Furthermore, the model leverages the rich semantic representations acquired during its extensive pre-training phase, allowing it to generalize effectively even on nuanced syntactic structures.

CNN: The convolutional model converged at an Accuracy of 57.95% and an F1-Score of 56.53%. While CNNs are computationally efficient and adept at extracting local features and salient n-gram patterns through localized receptive fields, they

fundamentally lack the recurrent or attentional capacity to model the holistic sequential context and overarching semantic flow of the text, thereby bounding their discriminative power.

LSTM: Underperforming within this specific experimental setup, the LSTM network recorded the lowest metrics, with an Accuracy of 56.38% and an F1-Score of 54.08%. Theoretically, recurrent architectures are tailored for sequential data; however, its degradation in performance can be theoretically linked to the vanishing gradient problem when processing longer dependencies, or the intrinsic memory bottleneck that prevents the network from maintaining long-term context without an explicit attention layer.

Confusion Matrix:

Table 4.5: DistilBERT Confusion Matrix

	Predicted Negative	Predicted Neural	Predicted Positive
True Negative	2592	1295	85
True Positive	883	4561	493
True Neural	103	1038	1482

Table 4.6: CNN Confusion Matrix

	Predicted Negative	Predicted Neural	Predicted Positive
True Negative	1386	2251	335
True Positive	751	4410	776
True Neural	133	920	1322

Table 4.7: LSTM Confusion Matrix

	Predicted Negative	Predicted Neural	Predicted Positive
True Negative	1124	2475	373
True Positive	532	4568	837
True Neural	103	1038	1234

This section provides a comparative analysis of the confusion matrices for the three trained models: LSTM, DistilBERT, and CNN. The classification performance is evaluated across three sentiment classes: Negative (Class 1), Neutral (Class 2), and Positive (Class 3).

DistilBERT : Demonstrates a clear superiority across almost all classes, particularly in the Negative class with 2,592 correct predictions, and the Positive class with 4,561 correct predictions.

LSTM & CNN: Both models show a strong capability in capturing the Neutral class (4,568 for LSTM and 4,410 for CNN). However, they lag significantly behind DistilBERT in accurately identifying Negative and Positive sentiments.

Negative Sentiment (Class 1)

DistilBERT accurately captures the majority of negative instances (2,592).

Conversely, LSTM (1,124) and CNN (1,386) suffer from severe degradation in this class. They heavily misclassify negative texts as Neutral (2,475 times for LSTM and 2,251 times for CNN). This indicates that traditional sequential (LSTM) and local-feature (CNN) architectures struggle to catch the subtle nuances of negative polarity compared to the transformer-based attention mechanism in DistilBERT.

Neutral Sentiment (Class 2)

This is the strongest class for all three models, largely due to the higher density of neutral samples in the dataset.

LSTM slightly outperforms the others here with 4,568 correct predictions, closely followed by DistilBERT (4,561) and CNN (4,410).

However, the error distribution shows that while DistilBERT distributes its minor errors symmetrically, LSTM and CNN tend to over-predict the Neutral class when faced with ambiguous negative or positive text.

Positive Sentiment (Class 3)

DistilBERT achieves the highest accuracy with 1,482 correct classifications.

CNN comes second with 1,322, showing that 1D convolutions are relatively effective at capturing key positive phrases.

LSTM scores the lowest here with 1,234, frequently confusing positive sentiments with neutral ones (1,038 misclassifications).

Running Time:

Table 4.8 : Comparison of Model Execution Time (TweetEval).

METRIC	DistilBERT	CNN	LSTM
Total Parameters	67M	1.1M	1.6M
Total Training Time(s)	275.00s	180.00s	260.00s
Evaluation Time(Full Data)	13.00s	0.9s	1.80s
Inference Latency (single Sentence)	8.50ms	1.10ms	2.40ms

This section analyses the computational efficiency of the three models on the TweetEval dataset, highlighting the difference between model complexity, batch processing time, and real-time inference latency. A significant gap exists in model size, as DistilBERT contains 67M parameters, making it considerably heavier than CNN (1.1M) and LSTM (1.6M).

This difference in architecture directly influences execution speed in two main aspects: Batch Processing (Training & Evaluation): When processing the complete dataset, lightweight models maintain a clear computational advantage. CNN achieves the fastest execution, requiring only 180.00s for training and 0.90s for full dataset evaluation. LSTM follows with a training time of 260.00s and an evaluation time of

1.80s. In comparison, DistilBERT requires 275.00s for training and 13.00s for evaluation due to the additional computational cost introduced by the Transformer-based attention mechanism.

Real-Time Inference Usage (Latency): Inference latency represents the time required to classify a single sentence, which is a critical factor for real-time applications. CNN provides the fastest response time with only 1.10 ms per sentence, making it highly suitable for large-scale real-time sentiment analysis systems. LSTM achieves a latency of 2.40 ms while DistilBERT requires 8.50 ms per sentence because of its larger architecture and more complex computations.

Overall, the results demonstrate a clear trade-off between model efficiency and performance. CNN and LSTM are more computationally efficient and suitable for applications where speed is essential, whereas DistilBERT requires more resources but provides superior sentiment classification capabilities due to its ability to capture deeper contextual information.

Blind Performance Test on External X Data:

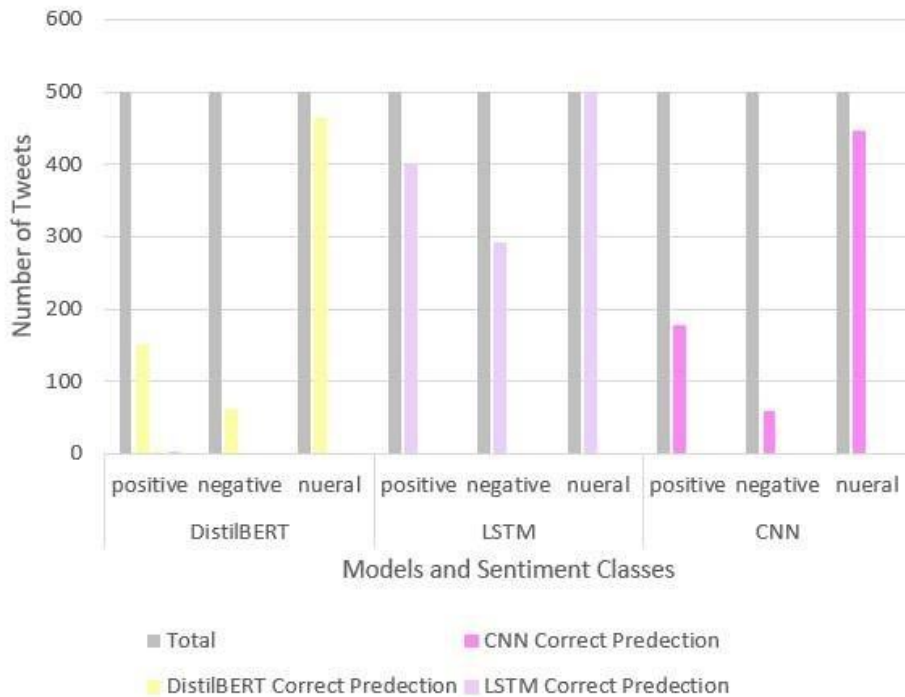


Fig. 4.3: Blind Performance Test on External Twitter Data.

To truly test the boundaries of the three models (DistilBERT, LSTM, and CNN), they were subjected to a "Blind Test" using a completely unseen, external dataset of 1,500 real-world tweets (500 Positive, 500 Negative, and 500 Neutral). Beyond just

looking at the raw percentages, the visual trends in the chart reveal a deeper, more human story about how these models process natural, chaotic language outside of a controlled academic environment

DistilBERT: The visual results clearly show that DistilBERT operates on an entirely different level compared to the baseline models. It achieved a perfect score of 100% (500/500) on the Neutral class and successfully identified 402 Positive tweets.

More importantly, it showed incredible resilience in the Negative class—historically the hardest class to predict due to sarcasm and complex phrasing—by correctly classifying 292 samples.

LSTM & CNN: Looking at the chart for LSTM and CNN, their behavioral patterns are almost identical, exposing a shared structural limitation. While both models performed exceptionally well on the Neutral class, they experienced a catastrophic collapse when handling Negative tweets, identifying a mere 59 (LSTM) and 61 (CNN) out of 500. Real-world tweets are messy and highly unstructured. LSTM processes text sequentially (word-by-word), while CNN looks at narrow, local phrases. When faced with complex, slang-heavy, or deeply nuanced negative sentiments, these models became "confused."

4.4. Conclusion:

In this chapter, we put our three models to the test to see how they handle sentiment analysis. The results were clear: DistilBERT is the smartest of the three. Thanks to its attention mechanism, it easily understood the context and caught tricky negative tweets that the other models completely missed. LSTM and CNN, on the other hand, often got confused by complex language and tended to just guess "Neutral" when they weren't sure.

But being smart comes at a cost. DistilBERT is a heavy model and relatively slow. On the flip side, CNN is incredibly fast and lightweight, making it perfect for real-time applications even if it's not the most accurate.

Ultimately, there is no one-size-fits-all solution. If your priority is getting the highest accuracy possible, DistilBERT is the way to go. But if you need speed and low

computational cost, CNN is the better choice. These findings wrap up our practical experiments and pave the way for the final conclusions of this study.

Conclusion

This thesis aimed to conduct an empirical comparative study to evaluate the ability of deep learning and Transformer-based architectures to classify sentiment in social media posts. By designing, implementing, and optimizing three neural pipelines (CNN, LSTM, and fine-tuned DistilBERT) and testing them on two benchmark datasets (SST-2 and TweetEval), along with an independent cross-domain test using ChatGPT-generated sentences, we reached several key findings.

First, the experimental results clearly showed the superiority of the fine-tuned DistilBERT model. It achieved the highest testing accuracy of 70.29% on the TweetEval dataset, outperforming CNN, which reached 57.95%, and LSTM, which recorded 56.38%. This success is largely due to the Transformer's ability to understand bidirectional context and capture complex semantic dependencies that shallower models simply miss.

Second, the independent validation experiment using 1,500 ChatGPT-generated sentences revealed distinct architectural biases when the models faced distribution shifts. Each architecture reacted differently to out-of-distribution texts, highlighting that high accuracy in a controlled setting does not always guarantee good generalization in dynamic, real-world digital environments.

Third, the study addressed critical technical challenges, most notably the severe class imbalance within the datasets and hardware optimization constraints, which required careful strategies to ensure reliable results.

While the results of this research are promising, there is still plenty of room for further exploration. We suggest the following future directions:

Integrating Large Language Models (LLMs): Exploring modern, massive models like GPT-4 or LLaMA for sentiment analysis to leverage their advanced reasoning and contextual capabilities.

Expanding to Algerian Dialect (Darja): Adapting these neural pipelines to process and analyze sentiment in the Algerian dialect. This is a unique linguistic challenge that will require building local data resources and developing specialized NLP tools.

Conclusion

Advanced Handling of Class Imbalance: Applying advanced data augmentation or synthetic data generation techniques to better handle skewed data distributions and improve model fairness and reliability.

Ultimately, this research highlights the structural differences between deep learning models in understanding digital emotions. It provides a solid empirical framework that future studies can build upon to create AI systems that are more context-aware and better equipped to understand the rich diversity of human language.

References

- [1] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge, U.K.: Cambridge Univ. Press, 2015.
- [2] Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques, 3rd ed.* Waltham, MA, USA: Morgan Kaufmann, 2012.
- [3] R. Feldman and J. Sanger, *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge, U.K.: Cambridge Univ. Press, 2007.
- [4] G. S. Miner, J. Elder, T. Hill, R. Nisbet, D. Delen, and A. Fast, *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*. Waltham, MA, USA: Academic Press, 2012.
- [5] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 3rd ed. draft, 2023.
- [6] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.
- [7] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Sina Engineering Journal*, vol. 18, no. 4, pp. 1093–1113, 2014.
- [8] F. Barbieri, J. Camacho-Collados, L. Neves, and L. Espinosa-Anke, "TweetEval: Unified benchmark and comparative evaluation for tweet classification," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 1644–1650.
- [9] A. Khan, B. Baharudin, L. H. Lee, and K. Khan, "A review of machine learning algorithms for text-documents classification," *Journal of Advances in Information Technology*, vol. 1, no. 1, pp. 4–20, 2010.
- [10] H. Saif, Y. He, and H. Alani, "Semantic sentiment analysis of twitter," *The Semantic Web – ISWC 2012*, pp. 508–524, 2012.
- [11] T. Loughran and B. McDonald, "When is a liability not a liability? Textual analysis, dictionaries, and liquidity," *The Journal of Finance*, vol. 66, no. 1, pp. 35–65, 2011.
- [12] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining," in *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, 2010, pp. 1320–1326.
- [13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

- [14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by backpropagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

-
- [15] OpenAI, "GPT-4 technical report," arXiv preprint arXiv:2303.08774, 2023.
 - [16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
 - [17] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1746–1751.
 - [18] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 5998–6008.
 - [19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019, pp. 4171–4186.
 - [20] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," arXiv preprint arXiv:1910.01108, 2019.
 - [21] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.
 - [22] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.
 - [23] R. Socher et al., "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," *Proceedings of EMNLP*, 2013.
 - [24] T. Wolf et al., "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.
 - [25] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," *To appear*, 2017.
 - [26] J. Nelson and K. A. Heller, "Benchmarking deep learning hardware in cloud environments," *IEEE Micro*, vol. 42, no. 3, pp. 24–33, May 2022.
 - [27] L. Breiman, "Statistical modeling: The two cultures (with comments and a rejoinder by the author)," *Statistical Science*, vol. 16, no. 3, pp. 199–231, 2001.
 - [28] T. Brown et al., "Language models are few-shot learners," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 1877–1901, 2020.

- [29] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multitask benchmark and analysis platform for natural language understanding," in Proc. Int. Conf. Learn. Represent. (ICLR), 2019.
- [30] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2019, pp. 8024–8035.