UNIVERSITY OF THE AEGEAN

DEPARTMENT OF INFORMATION AND COMMUNICATION SYSTEMS ENGINEERING (ICSD)

POSTGRADUATE STUDIES PROGRAMME

Internet of Things: Intelligent environments in next-generation networks

Market Surveillance from Product's Reviews using Semantic Similarity Measures, BERT, VADER, and Natural Language Processing

Theocharis Theocharidis

Supervisor: Mr. Prof. Dr. Panagiotis Symeonidis

Board of Examiners: Mr. Prof. Dr. Efstathios Stamatatos, Mr. Prof. Dr. Alexios Kaporis

Samos, July, 2024

This page is intentionally left white.

Preface

This dissertation addresses the effectiveness of market surveillance through user review analysis, the combination of semantic similarity (SemSim) measures with Natural Language Processing (NLP) techniques, and the performance of the proposed algorithms in real-world scenarios. It explores the development and application of algorithms leveraging semantic similarity measures and NLP techniques to analyze textual data, particularly user reviews and product evaluations. By quantifying the SemSim between words and phrases, these measures enable a deeper semantic understanding, facilitating the drawing of conclusions, crucial for evaluating product safety and suitability based on user feedback. This dissertation presents a comprehensive methodology, from data collection and preprocessing to the application of semantic similarity measures through proposed algorithms. The effectiveness of these algorithms is demonstrated through experiments on both synthetic and real-world datasets, specifically Amazon product reviews in the category 'Toys and Games'. The results reveal the strengths and limitations of different semantic similarity measures in categorizing and interpreting user reviews. Additionally, this study incorporates the use of advanced NLP models such as BERT and VADER, providing a comparative analysis of their performance alongside the proposed algorithms. Future directions for enhancing the algorithms and expanding their applicability to other domains are also outlined, including the integration of advanced Large Language Models (LLMs) and the Retrieval-Augmented Generation (RAG) technique to improve the accuracy and relevance of textual analysis.

Στο δυναμικό τοπίο των ψηφιακών πληροφοριών, η αποτελεσματική επεξεργασία και η εξαγωγή ουσιαστικών πληροφοριών από σύνολα δεδομένων αποκτά ολοένα και μεγαλύτερη σημασία. Η παρούσα διπλωματική ασχολείται με την αποτελεσματικότητα της εποπτείας της αγοράς μέσω της ανάλυσης των αξιολογήσεων των χρηστών, του συνδυασμού μέτρων Σημασιολογικής Ομοιότητας (ΣΟ) με τεχνικές επεξεργασίας φυσικής γλώσσας (NLP) και της απόδοσης των προτεινόμενων αλγορίθμων σε σενάρια πραγματικού κόσμου. Διερευνά την ανάπτυξη και εφαρμογή αλγορίθμων που αξιοποιούν μέτρα ΣΟ και τεχνικές NLP για την ανάλυση δεδομένων κειμένου, ιδίως κριτικές χρηστών και αξιολογήσεις προϊόντων. Με την ποσοτικοποίηση της ΣΟ μεταξύ λέξεων και φράσεων, τα μέτρα αυτά επιτρέπουν μια βαθύτερη σημασιολογική κατανόηση, διευκολύνοντας την εξαγωγή συμπερασμάτων που είναι κρίσιμης σημασίας για την αξιολόγηση της ασφάλειας και της καταλληλότητας των προϊόντων με βάση τα σχόλια των χρηστών. Παρουσιάζεται μια ολοκληρωμένη μεθοδολογία, από τη συλλογή και προεπεξεργασία δεδομένων, έως την εφαρμογή μέτρων ΣΟ μέσω προτεινόμενων αλγορίθμων. Η αποτελεσματικότητα αυτών των αλγορίθμων αποδεικνύεται μέσω πειραμάτων, τόσο σε συνθετικά όσο και σε πραγματικά σύνολα δεδομένων, συγκεκριμένα σε κριτικές προϊόντων Amazon στην κατηγορία "Toys and Games". Επιπλέον, αυτή η μελέτη ενσωματώνει τη χρήση προηγμένων μοντέλων NLP, όπως BERT και VADER, παρέχοντας μια συγκριτική ανάλυση της απόδοσής τους σε σύγκριση με τους προτεινόμενους αλγορίθμους. Προτείνονται μελλοντικές κατευθύνσεις για την ενίσχυση των αλγορίθμων και την επέκταση της εφαρμογής τους σε άλλους τομείς, όπως η ενσωμάτωση προηγμένων LLM και της τεχνικής Retrieval-Augmented Generation (RAG) για τη βελτίωση της ακρίβειας και της συνάφειας της ανάλυσης κειμένου.

Keywords: Semantic similarity, BERT, VADER, Natural Language Processing, Market surveillance, User reviews, Product safety, Large Language Models, Retrieval-Augmented Generation

Acknowledgements

During my postgraduate studies, I have been privileged to experience a high level of academic rigor and specialization. First and foremost, I would like to express my gratitude to all my professors who have generously shared their knowledge and expertise. Their commitment to excellence in teaching has equipped me with the essential skills and understanding necessary for my academic and professional development.

I am especially thankful to my supervising professor, Mr. Dr. Symeonidis, whose unwavering guidance and insightful feedback have been pivotal throughout the research process. His encouragement, attention to detail, and dedication have profoundly influenced my work, and I am deeply grateful for his mentorship.

Additionally, I would like to express my deepest appreciation to my family. Their constant support, understanding, and encouragement have been a source of great strength and motivation.

This page is intentionally left white.

Table of Contents

1 Introduction **1**
- 1.1 Research Questions . . . 2
- 1.2 Structure . . . 3

2 Literature Review **4**
- 2.1 EU Toy Safety Legislation . . . 4
- 2.2 Market Surveillance and Artificial Intelligence (AI) Applications . . . 5
- 2.3 Semantic Similarity Measures . . . 6
  - 2.3.1 Wu-Palmer Similarity (WUP) . . . 7
  - 2.3.2 Resnik Similarity (RES) . . . 8
  - 2.3.3 Jiang-Conrath Similarity (JCS) . . . 9
  - 2.3.4 Leacock-Chodorow Similarity(LCS) . . . 10
  - 2.3.5 Lin Similarity(LIN) . . . 11
  - 2.3.6 Path Similarity (PATH) . . . 11
  - 2.3.7 Cosine Similarity (COS) . . . 12
- 2.4 BERT (Bidirectional Encoder Representations from Transformers) . . . 13
- 2.5 VADER (Valence Aware Dictionary and Sentiment Reasoner) . . . 14

3 Proposed Methodology and Algorithms **15**
- 3.1 Toy Example . . . 15
- 3.2 Results of a Toy example Using Wu-Palmer Semantic Similarity Measure . . . 16
- 3.3 Methodology . . . 20
- 3.4 Algorithm for Detecting if a Toy is Dangerous Based on Users' Reviews . . . 21
  - 3.4.1 Pseudocode . . . 24
- 3.5 Advanced Algorithm for Detecting if a Toy is Dangerous . . . 25
  - 3.5.1 Symbols and Definitions . . . 26
  - 3.5.2 Algorithm Description . . . 27
  - 3.5.3 Pseudocode . . . 29
- 3.6 Ensemble Algorithm of different Semantics Similarity Measures (Boosting) . . . 31
  - 3.6.1 Symbols and Definitions . . . 31
  - 3.6.2 Algorithm Description . . . 32
  - 3.6.3 Pseudocode for Algorithm 3 . . . 35

4 Experimental Results **36**
- 4.1 Experimental Protocol . . . 36
- 4.2 Datasets . . . 37
  - 4.2.1 Synthetic Dataset . . . 37
  - 4.2.2 Amazon Reviews Dataset . . . 39
  - 4.2.3 Amazon Reviews Dataset . . . 40
- 4.3 Evaluation Metrics . . . 40
  - 4.3.1 Accuracy . . . 41
  - 4.3.2 Precision . . . 42
  - 4.3.3 Recall . . . 43
  - 4.3.4 ROC and AUC . . . 44
- 4.4 Experiment 1: Results of the Main Algorithm Using Various Semantic Similarity Measures . . . 47
  - 4.4.1 Application on Synthetic Dataset . . . 47
  - 4.4.2 Application on the Amazon Dataset . . . 51

4.5 Experiment 2: Results of the Advanced Algorithm using various Semantic Similarity Measures 53
4.5.1 Application on Synthetic Dataset . . . 54
4.5.2 Application on Amazon Dataset . . . 56
4.6 Experiment 3: Results of the Ensemble Algorithm (Boosting) using various Semantic Similarity Measures . . . 59
4.6.1 Application on Amazon Dataset . . . 59
4.7 Performance Evaluation of BERT and VADER Models . . . 63

5 Discussion and Future work 67
5.1 Future Work . . . 69
5.2 Algorithm for Classifying Products Based on User Reviews Using Large Language Models (LLMs) . . . 69
5.2.1 Detailed Explanation of the Algorithm . . . 70
5.2.2 Symbols and Definitions . . . 71
5.2.3 Pseudocode for the Algorithm 4 . . . 72
5.2.4 Experiment 4: Results of the Application of GPT-3.5 Turbo on Amazon Dataset . . . 72
5.3 Variation of Algorithm 4 Using Retrieval-Augmented Generation (RAG) . . . 73
5.3.1 Detailed Description of the Variation Using RAG . . . 74
5.3.2 Pseudocode for Algorithm 5 . . . 75
5.3.3 Application of GPT-3.5 Turbo with Retrieval-Augmented Generation on Amazon Dataset 76
5.3.4 Integration of Advanced LLMs . . . 77
5.4 Challenges and Limitations . . . 77

Bibliography 78

Appendix 82

List of Figures

1 Problem to be addressed . . . 15
2 Detection of the problem . . . 16
3 Solution . . . 16
4 Sentiment analysis of tweets for a toy example. . . . 18
5 Wordcloud of dangerous terms found in related reviews for the toy example. . . . 18
6 Danger-related words frequency for the toy example. . . . 19
7 Bar chart of semantic similarity values for each tweet in the toy example using Wu-Palmer measure. . . . 20
8 Filtering and Sentiment Analysis . . . 21
9 Semantic Similarity and Final Categorization . . . 22
10 Categorization and Vectorization . . . 25
11 Filtering and Creating Analysis Terms . . . 25
12 Processing and Normalizing . . . 25
13 Visual Representation of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) . . . 31
14 Confusion matrix illustrating the calculation of accuracy with True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). . . . 42
15 Confusion matrix illustrating the calculation of precision with True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). . . . 43
16 Confusion matrix highlighting the components relevant for calculating Recall. . . . 45
17 ROC curve illustrating the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR) for different threshold levels. . . . 46
18 Interpolated Precision-Recall Curves. . . . 48
19 Comparative ROC Curves. . . . 50
20 Comparative Precision-Recall Curve for the Amazon Dataset . . . 52
21 Comparative ROC Curve Diagram for the Amazon Dataset . . . 53
22 Comparative Precision-Recall Curves of the application of Advanced Algorithm on the Synthetic Dataset . . . 55
23 Comparative ROC Curves of the application of Advanced Algorithm on the Synthetic Dataset 56
24 Comparative Precision-Recall Curves of the application of Advanced Algorithm on the Amazon Dataset . . . 57
25 Comparative ROC Curves Diagram after the application of Advanced Algorithm on the the Amazon Dataset . . . 58
26 Interpolated Precision-Recall Curves after the application of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) on the Amazon Dataset . . . 62
27 Comparative ROC Curves Diagram after the application of Ensemble Algorithm of different Semantics Similarity Measures (Boosting) on the Amazon Dataset . . . 63
28 BERT Interpolated Precision-Recall Curve . . . 64
29 BERT ROC Curve . . . 64
30 VADER Interpolated Precision-Recall Curve . . . 65
31 VADER ROC Curve . . . 65
32 Comparative ROC Curves for Simple Algorithm, Advanced Algorithm, Ensemble Algorithm (Boosting), BERT, and VADER. The area under the curve (AUC) values are 0.81, 0.85, 0.96, 0.94, and 0.77 respectively, indicating the performance of each algorithm. The dashed line represents the performance of a random classifier (AUC = 0.50). . . . 68

33 Visual Representation of the Algorithm for Classifying Products Based on User Reviews Using LLMs 69
34 Precision-Recall Curve 73
35 ROC Curve GPT3.5 turbo 73
36 Visual Representation of the RAG Technique Variation for the Algorithm 73
37 Textwrap Visualization of Relevant Reviews for the Search Term using RAG 76

List of Tables

1 Users' Reviews Examples . . . 15
2 Relevant tweets extracted for the toy example. . . . 17
3 Example of the syntheric dataset of tweets simulating product reviews and safety concerns. . . . 38
4 Danger Related Terms . . . 39
5 Performance metrics for each semantic similarity measure in Synthetic dataset. . . . 48
6 Best metric values for each semantic similarity measure in the Amazon dataset. . . . 51
7 Performance metrics after the application of Advanced Algorithm for each semantic similarity measure in Synthetic dataset. . . . 54
8 Best metric values for each semantic similarity measure in the Amazon dataset using the Advanced Algorithm. . . . 56
9 Selection Menu for Amazon ASINs . . . 59
10 Maximum Semantic Similarity Scores . . . 60
11 Optimized Classes and Weights for Semantic Similarity Measures . . . 60
12 Best metric values for each semantic similarity measure in the Amazon dataset using the Ensemble Algorithm of different Semantics Similarity Measures (Boosting). . . . 61
13 Performance Metrics for all the algorithms, including BERT and VADER, on the Amazon dataset. . . . 64
14 Performance metrics of GPT-3.5 Turbo at various threshold levels on the Amazon dataset. . . . 72

List of Algorithms

1 Simple Algorithm for Detecting if a Toy is dangerous 24
2 Advanced Algorithm for Detecting if a Product is Dangerous 30
3 Ensemble Algorithm of different Semantics Similarity Measures (Boosting) 35
4 Algorithm for Classifying Products Based on User Reviews Using Large Language Models 72
5 Algorithm for Classifying Products Based on User Reviews Using Large Language Models with RAG Technique 75

List of Equations

1 Wu-Palmer Similarity . . . 8
2 Resnik Similarity . . . 8
3 Jiang-Conrath Similarity . . . 9
4 LCS Equation . . . 10
5 Lin Similarity . . . 11
6 Path Similarity . . . 12
7 Cosine Similarity . . . 13
8 Filtering Reviews . . . 22
9 Sentiment Analysis . . . 23
10 Preprocessing . . . 23
11 Semantic Similarity Calculation . . . 23
12 Weighting and Averaging . . . 23
13 Final Categorization . . . 23
14 Categorize Reviews . . . 27
15 Vectorization . . . 27
16 Filter Words Not in Vocabulary . . . 27
17 Create Analysis Terms List . . . 27
18 Process Analysis Terms with Sentiment Analysis . . . 27
19 Normalize Similarity Scores . . . 28
20 Classes Creation . . . 32
21 Frequency Calculation for Each Class . . . 33
22 Exponential Weight Calculation . . . 33
23 Weight Normalization . . . 33
24 Threshold Calculation . . . 33
25 Proportion of Scores Below Threshold . . . 33
26 Adjustment of Parameter a . . . 34
27 Weight Adjustment . . . 34
28 Building Headers and Data . . . 70
29 Creating Prompt . . . 70
30 Sending Prompt and Receiving Response . . . 70
31 Processing Output . . . 70
32 Aggregating Outputs . . . 70
33 Returning Output . . . 71
34 Retrieving Additional Information . . . 74
35 Retrieving Additional Information . . . 74
36 Sending Prompt with Additional Information and Receiving Response . . . 75
37 Processing Output with Additional Information . . . 75

1 Introduction

In an era where digital information burgeons exponentially, the quest for efficient and accurate mechanisms to sift through, analyze, and interpret vast datasets becomes increasingly paramount. This dissertation delves into the development and application of a core algorithm and subsequent variations that leverage semantic similarity measures and Natural Language Processing (NLP) techniques to extract insights from textual data. The focal point of this investigation is the realm of user reviews and product or service evaluations, a domain where the sheer volume of data presents both a challenge and an opportunity for comprehensive analysis.

Semantic similarity measures offer a sophisticated avenue to gauge the closeness of concepts within linguistic models. By quantifying the degree of relatedness between sets of words or phrases, these measures enable the algorithm to discern nuanced patterns and trends in textual content. The application of such measures extends beyond mere syntactical analysis, venturing into the semantic essence of language—a realm where words are imbued with meaning and context.

Moreover, the inclusion of advanced sentiment analysis models such as BERT (Bidirectional Encoder Representations from Transformers) and VADER (Valence Aware Dictionary and Sentiment Reasoner) further strengthens the analytical framework. BERT, a deep learning model, excels at understanding the context and nuances of language, thereby improving the accuracy of sentiment classification. VADER, on the other hand, is a lexicon and rule-based model that provides efficient sentiment analysis, particularly effective in social media texts.

Additionally, LLMs, such as OpenAI's GPT and its successors, represent a significant advancement in the field of NLP. These models, trained on vast amounts of data, have demonstrated an unparalleled ability to understand and generate human-like text. By integrating LLMs, this dissertation aims to enhance the capability of the algorithm to classify and interpret user reviews with greater accuracy and depth.

To address the aforementioned goals, the core objective of the algorithms developed are to predict the suitability of a product or service based on user-generated content. They endeavor to classify a product as safe or dangerous, suitable or unsuitable by analyzing the sentiment and thematic content of user reviews. This process involves several key steps:

- Collection and preprocessing of user reviews to extract the primary lemmas, thereby distilling the essence of the textual data.
- Construction of a lexicon or list of terms that serve as benchmarks for categorizing products or services.
- Application of semantic similarity measures to compare each lemma in the review against the curated list of terms, thereby generating a semantic similarity score for each comparison.
- Aggregation of these scores to formulate a composite metric that reflects the overall sentiment and thematic alignment of the review with the characteristics of the product or service under scrutiny.
- Integration of BERT and VADER models to enhance the sentiment analysis component, leveraging their respective strengths in deep learning and rule-based sentiment assessment.
- Utilization of LLMs, as a future work, to enhance the analysis by providing context-aware interpretations and classifications of the reviews.

More specifically, this dissertation introduces a variety of semantic similarity measures, including Wu-Palmer, Resnik, Jiang-Conrath, Lin, Path, Leacock-Chodorow, and Cosine similarity. Each of these measures contributes a unique perspective on semantic analysis, collectively enriching the algorithm's ability to interpret and classify textual data accurately. Additionally, the incorporation of BERT and VADER models enhances the algorithm's capability to perform advanced sentiment analysis, capturing both deep contextual understanding and efficient rule-based assessments.

An illustrative proof of concept, centered around the classification of products as dangerous or safe based on the European Union's toy safety legislation, underscores the algorithm's potential. This initial exploration paves the way for broader applications, ultimately leading to an evaluation of the algorithm's performance using a real-world dataset of Amazon product reviews.

The methodology chapter meticulously outlines the algorithm's design and operational framework, detailing each step of the process from data collection to semantic analysis and classification. The successive chapters will delve into the results of the algorithm's application to various datasets, critically analyzing its effectiveness and pinpointing areas for refinement.

In summary, this dissertation presents a comprehensive exploration of an algorithm that stands at the intersection of semantic similarity measures,BERT, VADER, LLMs, and NLP. Through the lens of user reviews, it showcases the potential of semantic analysis and advanced language models to uncover insights that lie beneath the surface of textual data, offering a glimpse into the future of data analysis in the digital age.

This introduction sets the stage for a detailed examination of the algorithm's foundations, its application to real-world data, and the insights gleaned from its deployment. Through the subsequent chapters, the dissertation will unravel the complexities of semantic similarity measures and their pivotal role in enhancing our understanding of textual content.

1.1 Research Questions

This section defines the research questions that are the focus of this dissertation. These questions are aimed at guiding the investigation and providing a clear framework for understanding the scope and objectives of the research.

- **RQ1: How can market surveillance be effectively conducted through the analysis of user reviews?**

  This research question seeks to explore methods for leveraging user reviews to perform effective market surveillance, ensuring product safety and compliance.

- **RQ2: How can user reviews be effectively analyzed to identify the safety and suitability of products, and how do semantic similarity measures and NLP techniques enhance this analysis?**

  The goal of this question is to examine techniques for analyzing user reviews to determine product safety and suitability for consumers, and to investigate the integration of semantic similarity measures with NLP techniques to improve the accuracy and depth of user review analysis.

- **RQ3: How do the proposed algorithms and advanced NLP models, such as BERT and VADER, perform in real-world scenarios for market surveillance based on user reviews?**

The aim is to evaluate the effectiveness and applicability of the proposed algorithms and advanced NLP models like BERT and VADER in real-world market surveillance scenarios, assessing their performance and comparing their strengths and weaknesses.

1.2 Structure

This dissertation is structured into several chapters, each focusing on different aspects of the research, from theoretical foundations to practical applications and results. The chapters are outlined as follows:

The second chapter, Literature Review, presents the related work that forms the foundation of this thesis. It begins with an exploration of semantic similarity measures and their importance in Natural Language Processing (NLP) and semantic analysis. This chapter delves into various semantic similarity measures, such as Wu-Palmer, Resnik, Jiang-Conrath, Lin, Path, Leacock-Chodorow, Cosine similarity, and TF/IDF. Each measure is discussed in detail, highlighting its methodology, applications, and significance in understanding the nuances of language. Additionally, the chapter briefly introduces advanced NLP models such as BERT and VADER, describing their mechanisms and showcasing their applications in fields like sentiment analysis based on relevant literature.

The next chapter, Proposed Methodology and Algorithms, outlines the methodology and algorithms developed to analyze user reviews for product safety assessment. This chapter includes a toy example to illustrate the problem and solution. It describes the main algorithm for detecting dangerous products, emphasizing the improvements in categorization, preprocessing, vectorization, and normalization techniques. These methodologies provide a comprehensive framework for the detection and analysis of dangerous products based on user reviews.

Following the methodology, the Experimental Results chapter describes the experimental protocol and presents the results of testing the algorithms on various datasets, including synthetic and real-world datasets like Amazon product reviews. This chapter includes detailed analyses of the algorithms' performance, supported by graphs, tables, and commentary. The results demonstrate the effectiveness of the proposed methodology and the impact of different semantic similarity measures, BERT and VADER, on classification accuracy. The experimental findings provide valuable insights into the strengths and limitations of the developed algorithms.

The Discussion chapter provides a comprehensive summary of the findings and their implications. It synthesizes the results from the experimental analyses and discusses the key contributions and insights gained from the research. This chapter also explores potential extensions and future directions for improving and expanding the current work. By reflecting on the outcomes and their significance, the discussion offers a holistic understanding of the research's impact on product safety assessment through semantic analysis and machine learning.

The dissertation concludes with the Bibliography, which lists all the references and sources cited throughout the research.

Lastly, the Appendix includes important parts of the code used to implement the algorithms. This section provides detailed technical information for readers interested in replicating or extending the study, ensuring transparency and reproducibility in the research process. The whole code is available in the following GitHub repository: `https://lmy.de/jjdwj` .

2 Literature Review

Product safety is a paramount concern for consumers, as it directly impacts their health and well-being. Among various product categories, toys hold a significant place due to their widespread use by children. Ensuring the safety of toys is particularly crucial, given the vulnerability of the young users. To address these concerns, the European Union has established stringent regulations aimed at safeguarding toy safety. This comprehensive legislative framework is designed to minimize risks and protect children from potential hazards associated with toys. The importance of product safety extends beyond toys, encompassing various consumer goods where safety concerns can lead to serious health risks. Effective market surveillance and regulatory frameworks are critical in mitigating these risks and ensuring consumer safety. Various studies have highlighted the significance of robust safety measures and the role of technology in enhancing market surveillance processes [1, 2].

2.1 EU Toy Safety Legislation

The European Union's toy safety legislation is detailed and robust, aiming to ensure that all toys marketed within the EU meet specific safety standards. The key document governing this is the Directive 2009/48/EC, commonly known as the Toy Safety Directive [3]. This directive outlines various safety requirements that toys must comply with before they can be sold in the EU market. These requirements cover a wide range of safety aspects, including physical, mechanical, chemical, and electrical properties of toys, ensuring they are safe for children to use.

The European Union's Toy Safety Directive 2009/48/EC sets out stringent requirements to ensure toys' safety and protect children's health. This directive mandates that all toys sold within the EU must meet specific safety standards before entering the market. These standards cover various aspects of toy safety, including mechanical and physical properties, flammability, chemical properties, electrical properties, hygiene, and radioactivity [3]:

- **Physical and Mechanical Properties:** Ensuring toys do not present choking hazards, sharp edges, or small parts that could be swallowed.
- **Flammability:** Toys must not be highly flammable and should not present a fire hazard.
- **Chemical Properties:** Limiting the use of hazardous substances and chemicals that could be toxic, carcinogenic, or otherwise harmful.
- **Electrical Properties:** Ensuring electrical toys are designed safely to prevent electric shocks or burns.
- **Hygiene:** Toys should be hygienic and not pose any risk of infection, sickness, or contamination.
- **Radioactivity:** Toys should not contain radioactive elements that could expose children to harmful radiation.

The directive emphasizes the responsibilities of manufacturers, importers, and distributors in ensuring compliance with safety standards. Manufacturers must carry out a safety assessment of their toys, compile technical documentation, and issue an EU declaration of conformity. Additionally, toys must bear the CE marking to indicate compliance with the directive's requirements. To further ensure toy safety, the directive includes

provisions for market surveillance by member states. This involves monitoring and controlling toys placed on the market, conducting sample checks, and taking appropriate measures to ensure non-compliant toys are withdrawn or recalled.

Research has shown that the automated identification of national implementing measures (NIMs) of European directives using text similarity techniques has proven to be effective. Techniques such as unsupervised lexical and semantic similarity, vector space models, latent semantic analysis, and topic models have been used to identify and evaluate national implementations of EU directives, including the Toy Safety Directive. These methods facilitate the monitoring of compliance across different member states, ensuring that national laws align with the EU's regulatory framework [4].

Moreover, various studies have highlighted the role of technology in enhancing the efficiency of market surveillance. For example, the use of machine learning and natural language processing techniques can significantly improve the identification and analysis of non-compliant products. By leveraging these technologies, regulatory authorities can better enforce safety standards and protect consumers from potential hazards [5].

2.2 Market Surveillance and Artificial Intelligence (AI) Applications

Continuous research is being conducted to enhance market surveillance methods to ensure product safety. Traditional approaches to market surveillance involve physical inspections, testing, and conformity assessments. However, with the advent of advanced technologies, new methodologies are being explored to improve the efficiency and effectiveness of market surveillance. Market surveillance is essential for ensuring that products on the market comply with safety standards and regulations. Traditional surveillance methods have relied heavily on physical inspections and product testing, which can be time-consuming and resource-intensive [6].

Artificial Intelligence (AI) has emerged as a powerful tool in this context. AI technologies, particularly those involving machine learning and natural language processing, have shown great potential in monitoring and analyzing vast amounts of data. These technologies can assist in identifying unsafe products, detecting non-compliance with safety standards, and predicting potential hazards before they cause harm [7]. For example, machine learning classifiers have been successfully used to detect wildlife product promotion and sales on social media platforms, demonstrating the potential of AI in market surveillance [8].

One of the promising applications of AI in market surveillance is sentiment analysis. Sentiment analysis involves using natural language processing to analyze customer reviews and feedback to determine their sentiment, which can be positive, negative, or neutral. This method can be similarly applied to monitor toy safety by analyzing customer reviews and online discussions about toys. Moreover, sentiment analysis has been used to understand market trends and consumer preferences, making it a valuable tool for market surveillance [9].

Additionally, advanced text mining techniques such as Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (pLSA) have been used to extract meaningful information from product reviews. These techniques help in summarizing reviews and identifying common safety-related issues reported by consumers [10]. Other approaches, such as word embeddings and neural network-based models, have been utilized to measure semantic similarity and improve the efficiency of market surveillance by identifying related content across large datasets [11].

In the domain of food safety, text mining approaches have also been applied to postmarket surveillance. Gold-

berg et al. [12] demonstrated the use of text mining and machine learning to rapidly screen online media for mentions of food safety hazards, achieving high accuracy in identifying risky products.

Moreover, the integration of semantic similarity techniques in market surveillance has shown promising results. For example, Jia Liu and Olivier Toubia [13] developed a semantic approach for estimating consumer content preferences from online search queries, which can be adapted for monitoring consumer behavior and identifying potential safety concerns in various product categories.

Furthermore, the use of machine learning classifiers in market surveillance has expanded beyond traditional applications. For example, a study by Visweswaran et al. [14] evaluated various machine learning classifiers to identify vaping-related content on Twitter. The classifiers demonstrated high accuracy in distinguishing relevant tweets, commercial content, and sentiment, underscoring the potential of AI in real-time monitoring of social media platforms for public health surveillance.

Advanced frameworks combining multiple AI techniques have been proposed to improve surveillance efficiency. For instance, Atoum [15] introduced a framework that integrates semantic similarity and sentiment analysis to measure software quality-in-use based on user reviews. This approach can be adapted to market surveillance by applying similar techniques to product reviews and safety reports, thereby enhancing the detection of non-compliance and safety issues.

In this dissertation, it is explored the application of semantic similarity measures, a branch of AI, in market surveillance. Semantic similarity measures can be used to analyze textual data, such as product reviews and safety reports, to identify and categorize safety-related issues. By leveraging these measures, we aim to enhance the capability of market surveillance systems in ensuring product safety, particularly in the context of toys.

2.3 Semantic Similarity Measures

In Natural Language Processing (NLP) and semantic analysis, the concept of semantic similarity measures stands as a cornerstone, offering a systematic approach to quantify the conceptual closeness between words, phrases, or documents. These measures are instrumental in a multitude of applications, including but not limited to, information retrieval, text summarization, question answering, and especially in tasks where understanding the subtle nuances of language plays a critical role. The essence of semantic similarity measures lies in their ability to discern the degree of relatedness between semantic entities, thereby enabling algorithms to process and interpret human language in a manner that mirrors human comprehension.

Semantic similarity measures can be broadly categorized into three main types: lexical, corpus-based, and knowledge-based measures [9]. Lexical measures typically rely on the analysis of word forms and their co-occurrence patterns within a given corpus. Corpus-based measures, on the other hand, use statistical models derived from large text corpora to determine similarity. Knowledge-based measures utilize structured lexical databases such as WordNet to capture semantic relationships.

Takano and Kajikawa [16] explored the use of text similarity measures to identify commercialization opportunities in the Internet of Things (IoT) by analyzing massive quantities of papers and patents. Their work underscores the importance of semantic similarity in detecting technological trends and opportunities in rapidly evolving fields.

Hussain et al. [17] proposed a method for measuring semantic similarity between Wikipedia concepts using multiple inheritances. By defining the semantic space of a category and aggregating the semantic contribution weights of its relevant multiple ancestors, they demonstrated a significant improvement over traditional methods. This approach exemplifies the potential of using rich, structured knowledge sources to enhance semantic similarity measures.

In the context of market surveillance, semantic similarity measures have been employed to analyze large datasets and improve monitoring systems. For instance, Hain et al. [18] developed a scalable approach to measuring technological similarity between patents using embedding techniques from natural language processing. This method enables the representation of the entire patent universe as a technological network, facilitating the identification of knowledge flows and mapping technological change.

Furthermore, Kim and Sohn [19] introduced a machine-learning-based deep semantic analysis framework for forecasting technology convergence. By combining semantic analysis with traditional methods like link prediction and bibliometric analysis, they effectively predicted new technology fusions, showcasing the applicability of semantic similarity in anticipating future technological developments.

In practical applications, Venkatraman et al. [20] utilized a semantic similarity approach to classify spam emails in the Internet of Things (IoT) environment. By integrating Naïve Bayesian classification with conceptual and semantic similarity techniques, they achieved high accuracy in spam detection, highlighting the robustness of semantic similarity measures in diverse contexts.

Semantic similarity measures also play a crucial role in recommendation systems. Riyahi and Sohrabi [21] developed a hybrid recommender system for discussion groups based on implicit ratings and semantic similarity. Their system uses the semantic relevance of tags to provide more accurate recommendations, demonstrating the utility of semantic similarity in enhancing user experience and engagement in collaborative learning environments.

Lastly, Hernández-Castañeda et al. [22] presented a method for extractive automatic text summarization based on lexical-semantic keywords. By leveraging semantic information to improve keyword detection and clustering sentences to identify main topics, their approach improved the precision and coverage of generated summaries.

The following chapter delves into the intricacies of various semantic similarity measures, each with its unique methodology and application domain, starting with the Wu-Palmer similarity measure.

2.3.1 Wu-Palmer Similarity (WUP)

The Wu-Palmer Similarity measure, based on the work of Zhibiao Wu and Martha Palmer [23], is a widely recognized metric in the field of semantic similarity. It operates within the framework of a taxonomical structure, typically a lexical database like WordNet, where concepts are organized in a hierarchical manner. The Wu-Palmer metric computes the similarity between two synsets (sets of cognitive synonyms) by considering their depths within the hierarchy, as well as the depth of their least common subsumer (LCS) - essentially, the closest common ancestor in the taxonomy.

The Wu-Palmer similarity is calculated as the ratio of twice the depth of the LCS to the sum of the depths of

the two synsets in question and their LCS. Mathematically, it is expressed as:

$$\text{WUP}(synset_1, synset_2) = \frac{2 \times \text{depth}(LCS)}{\text{depth}(synset_1) + \text{depth}(synset_2) + 2 \times \text{depth}(LCS)} \tag{1}$$

This formula yields a similarity score that ranges between 0 and 1, where a score of 1 indicates complete semantic similarity, i.e., when the two synsets are identical, and a score closer to 0 suggests a lack of similarity. The Wu-Palmer metric is particularly praised for its intuitive interpretation and its emphasis on the informative richness of the LCS. By accounting for the hierarchical structure of the lexical database, the Wu-Palmer measure offers a nuanced assessment of similarity that is sensitive to the specific positions of concepts within a broader semantic context.

One of the strengths of the Wu-Palmer similarity is its balance between specificity and generality. It recognizes the importance of a shared context in determining semantic relatedness, making it particularly useful in applications where the hierarchical organization of concepts plays a pivotal role. However, it is also worth noting that the effectiveness of the Wu-Palmer similarity can be contingent upon the comprehensiveness and structure of the underlying lexical database, with its performance potentially varying across different domains or languages.

In summary, the Wu-Palmer similarity measure stands as a robust and insightful tool in the arsenal of semantic similarity measures, offering a meaningful quantification of conceptual relatedness grounded in the inherent structure of human language knowledge. As we progress through this chapter, we will explore additional measures, each contributing uniquely to the rich tapestry of semantic analysis methodologies.

2.3.2 Resnik Similarity (RES)

The Resnik Similarity measure [24] is another pivotal metric in the realm of semantic similarity, introduced by Philip Resnik. This metric distinguishes itself by focusing on the notion of shared information content between two concepts, which is quantified based on their least common subsumer (LCS) within a taxonomic structure like WordNet. Unlike other similarity measures that might consider the path lengths or depths of the concepts within the hierarchy, Resnik Similarity emphasizes the informational content shared by the concepts, rooted in their commonality.

The core idea behind Resnik Similarity is that the semantic relatedness of two concepts can be effectively captured by the amount of information needed to state what they have in common. Mathematically, Resnik Similarity is defined as the information content of the LCS of the two concepts, without considering the individual information contents of the concepts themselves. The formula for Resnik Similarity is given as:

$$\text{Resnik}(synset_1, synset_2) = -\log p(LCS(synset_1, synset_2)) \tag{2}$$

where $p(LCS(synset_1, synset_2))$ represents the probability of encountering an instance of the LCS in a specific corpus. This probability is inversely related to the information content, thus making rarer concepts (which are usually more specific and informative) yield higher similarity scores.

One of the main advantages of Resnik Similarity is its ability to provide a more nuanced understanding of semantic relatedness by quantifying the actual information shared between concepts. This makes it particularly useful in applications involving semantic disambiguation, where the goal is to determine the meaning of words in context.

However, the performance and applicability of Resnik Similarity can be influenced by the choice of corpus used to determine the information content of concepts, as well as the completeness and structure of the taxonomic database. Despite these considerations, Resnik Similarity remains a widely used and highly regarded metric for evaluating semantic similarity, offering valuable insights into the nature of conceptual relatedness within linguistic and cognitive frameworks.

Incorporating Resnik Similarity into semantic analysis tasks allows for a deeper exploration of the relationships between concepts, enriching our understanding of language and cognition through the lens of shared information content. In applications, Resnik Similarity facilitates a deeper understanding of the connections between concepts, enhancing linguistic and cognitive studies with its focus on informational commonality.

2.3.3 Jiang-Conrath Similarity (JCS)

The Jiang-Conrath Similarity (JCN) measure represents an innovative approach in the computation of semantic similarity between two concepts, leveraging the concept of information content to evaluate the distance between nodes in a semantic network such as WordNet. Proposed by Jay J. Jiang and David W. Conrath [25], this metric quantifies the dissimilarity between concepts, which is inversely related to their semantic similarity.

Central to the Jiang-Conrath approach is the notion that the semantic distance between two concepts can be effectively represented by the information content necessary to state their differences, rather than their commonalities. This perspective led to the development of a formula that contrasts sharply with other similarity measures focusing on shared information content. The Jiang-Conrath similarity is mathematically defined as:

$$JCN(synset_1, synset_2) = \frac{1}{IC(synset_1) + IC(synset_2) - 2 \times IC(LCS(synset_1, synset_2))} \tag{3}$$

where IC denotes the information content of a synset, and $LCS(synset_1, synset_2)$ is the least common subsumer of the two synsets. The information content is typically derived from the negative logarithm of the probability of encountering a synset in a large, representative corpus.

The distinctive characteristic of the Jiang-Conrath similarity lies in its interpretation of semantic distance. By calculating the inverse of the sum of the individual information contents minus twice the information content of their LCS, it offers a measure of how much information is exclusive to each of the concepts compared to what they share. As a result, lower values indicate greater similarity, with the metric adeptly capturing the nuances of concept specificity and generality.

The JCN similarity finds application in a variety of linguistic and cognitive tasks, including word sense disambiguation, ontology mapping, and semantic search. Its reliance on information content makes it particularly effective in contexts where detailed knowledge about concept specificity is crucial. However, the choice of corpus for calculating information content and the structure of the semantic network can significantly influence

the outcome, necessitating careful selection to ensure the relevance and accuracy of the similarity assessments.

In conclusion, the Jiang-Conrath similarity provides a compelling tool for the analysis of semantic relationships, offering insights that complement and enhance the broader spectrum of semantic similarity measures. Its focus on semantic distance through the lens of information content contributes a valuable perspective to the study of conceptual similarity and relatedness.

2.3.4 Leacock-Chodorow Similarity(LCS)

The Leacock-Chodorow Similarity [26], often abbreviated as LCS, offers a distinct approach to measuring the semantic similarity between two words or concepts within a hierarchical taxonomy such as WordNet. Introduced by Claudia Leacock and Martin Chodorow, this metric is based on the shortest path that connects the concepts in the taxonomy, adjusted by the maximum depth of the taxonomy itself.

The LCS similarity is calculated using the following formula:

$$LCS(synset_1, synset_2) = -\log\left(\frac{\text{length}(synset_1, synset_2)}{2 \times D}\right) \tag{4}$$

where length$(synset_1, synset_2)$ represents the shortest path between the two synsets in the taxonomy, and D denotes the maximum depth of the taxonomy. The logarithm of the path length divided by the maximum depth ensures that the similarity score decreases as the path length increases, reflecting the intuition that the further apart two concepts are in the taxonomy, the less similar they are.

This similarity measure is particularly insightful for applications requiring an understanding of conceptual distance within a well-defined hierarchy. The logarithmic scale used by the LCS method ensures that the similarity scores are normalized, allowing for meaningful comparisons across different parts of the taxonomy.

One of the key advantages of the Leacock-Chodorow similarity is its consideration of the taxonomy's depth, providing a more nuanced view of concept proximity than measures relying solely on path length. This feature makes it especially suitable for use in complex hierarchies where depth can vary significantly across branches.

However, the performance and relevance of the LCS similarity can be influenced by the specific characteristics of the taxonomy used, including its depth and the density of interconnections among concepts. Therefore, while LCS offers valuable insights into semantic similarity, its application should be tailored to the context of the taxonomy and the nature of the concepts being compared.

In summary, the Leacock-Chodorow Similarity contributes an important perspective to the analysis of semantic relationships, enabling a deeper understanding of conceptual proximity within hierarchical structures. Its application across linguistic and cognitive research underscores its utility in exploring the dimensions of semantic space.

2.3.5 Lin Similarity(LIN)

The Lin Similarity measure is an advanced metric for evaluating the semantic similarity between two words or concepts within an information-rich taxonomy, such as WordNet. Developed by Dekang Lin [27], this similarity metric is distinctive because it not only considers the hierarchical structure of the taxonomy but also incorporates the concepts' information content. The information content of a concept is determined by its frequency of occurrence in a large corpus, underpinning the assumption that less frequent concepts convey more information.

The formula for calculating Lin Similarity is given as:

$$Lin(synset_1, synset_2) = \frac{2 \times IC(LCS(synset_1, synset_2))}{IC(synset_1) + IC(synset_2)} \tag{5}$$

In this equation, IC represents the information content of a synset, and $LCS(synset_1, synset_2)$ denotes the least common subsumer of the two synsets, which is the most specific ancestor common to both synsets in the taxonomy. The Lin similarity score ranges from 0 to 1, where a score of 1 signifies complete similarity (i.e., the synsets are identical) and a score closer to 0 indicates lesser similarity.

One of the strengths of Lin Similarity is its emphasis on the informational aspect of semantic relatedness. By weighting the similarity score with the information content of the concepts, it ensures that the measure is sensitive to the specificity of the concepts being compared. This property makes Lin Similarity particularly useful for tasks that require a fine-grained understanding of semantic proximity, such as word sense disambiguation and semantic search.

However, the dependency on the information content also implies that the performance of the Lin Similarity measure can be affected by the choice of corpus used for calculating information content. Variations in corpus content and size may lead to differences in the calculated similarity scores.

In conclusion, Lin Similarity offers a sophisticated approach to quantifying semantic similarity, combining the structural insights of taxonomy-based measures with the richness of corpus-based information content. As we continue to explore semantic similarity measures, the Lin Similarity measure highlights the diverse methodologies available for capturing the nuances of semantic relationships.

2.3.6 Path Similarity (PATH)

Path Similarity is a fundamental metric in semantic analysis that quantifies the semantic proximity between two concepts based on the shortest path that connects their positions in a hierarchical taxonomy, such as WordNet[28]. Unlike other semantic similarity measures that incorporate information content or the depth of concepts within the taxonomy, Path Similarity solely relies on the path length, making it a straightforward yet effective approach to gauging semantic relatedness.

The essence of Path Similarity lies in its simplicity; it is calculated as the inverse of the shortest path length between two synsets (sets of cognitive synonyms) within the taxonomy:

$$PathSimilarity(synset\_1, synset\_2) = \frac{1}{shortestPathLength(synset\_1, synset\_2) + 1} \quad (6)$$

This formula ensures that the similarity score ranges between 0 and 1, where a score of 1 signifies identical synsets or concepts directly linked within the taxonomy, and a score approaching 0 indicates that the synsets are distantly related or not connected through a direct path. The addition of 1 in the denominator prevents division by zero and ensures that the formula can handle cases where the two synsets are the same, resulting in a path length of 0.

Path Similarity's reliance on path length alone makes it particularly appealing for applications requiring a quick and intuitive measure of semantic distance without the need for extensive computational resources. It is well-suited for tasks such as initial filtering of semantically related concepts, exploration of taxonomical structures, and in educational tools designed to illustrate the relationships between concepts[29].

However, the simplicity of Path Similarity can also be seen as a limitation. By not accounting for the depth of the synsets within the taxonomy or the information content derived from corpus statistics, Path Similarity may overlook the nuances of semantic richness and specificity. For instance, it treats all connections within the taxonomy equally, regardless of whether they span across general or highly specific concepts.

Despite these limitations, Path Similarity remains a valuable tool in the semantic similarity toolkit. Its intuitive interpretation and computational efficiency make it an indispensable metric for preliminary semantic analysis and educational purposes. As we delve further into semantic similarity measures, it becomes evident that each metric offers unique insights, with Path Similarity providing a fundamental baseline for understanding the concept of semantic distance.

In summary, Path Similarity exemplifies a direct and intuitive approach to measuring semantic relatedness, serving as a foundational metric that complements more complex measures in the analysis of semantic structures. Its application across various domains underscores the versatility and enduring relevance of path-based measures in semantic similarity assessment.

2.3.7 Cosine Similarity (COS)

Cosine Similarity is a measure used in the field of Information Retrieval and Text Mining to assess how similar two documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space[30]. In the context of text analysis, these vectors represent the Term Frequency-Inverse Document Frequency (TF-IDF) vectors of the documents.

The fundamental principle behind Cosine Similarity is that documents are considered similar if their vectors are close to each other in the vector space, which is indicated by a smaller angle between them. Conversely, a larger angle represents documents that are less similar. The cosine value ranges from -1 to 1, where 1 signifies identical orientation (maximum similarity), 0 indicates orthogonality (no similarity), and -1 represents completely opposite orientations.

The formula for computing Cosine Similarity between two vectors A and B is given by:

$$CosineSimilarity(A, B) = \frac{A \cdot B}{\|A\| \times \|B\|} \tag{7}$$

where $A \cdot B$ is the dot product of the vectors A and B, and $\|A\|$ and $\|B\|$ are the Euclidean norms of the vectors.

In practical terms, to calculate the Cosine Similarity between two text documents, one first transforms the texts into their TF-IDF representations, where each dimension corresponds to a unique term in the document corpus, and the values represent the relevance of each term to the document. The Cosine Similarity is then computed as the dot product of the TF-IDF vectors divided by the product of their norms[31].

This similarity measure is particularly useful in systems that recommend content, cluster documents, or detect duplicates by enabling a quantitative assessment of document similarity based on content. It is favored in scenarios where the magnitude of the vector (i.e., the length of the document) is not relevant, allowing for the comparison of documents of varying lengths on an equal footing.

However, while Cosine Similarity is a powerful tool for assessing textual similarity, it does not account for the semantic meaning of the words in the documents. Two documents could have high cosine similarity if they share a lot of terms, even if they discuss different topics. Therefore, it is often used in conjunction with other measures that consider semantic relatedness to provide a more comprehensive analysis.

2.4 BERT (Bidirectional Encoder Representations from Transformers)

BERT, developed by Google, represents a significant advancement in natural language understanding tasks through its transformer-based architecture. Unlike traditional models that process text sequentially, BERT leverages bidirectionality by considering both left-to-right and right-to-left contexts, thereby capturing the full context of each word based on its surroundings. This capability allows BERT to excel in various NLP tasks, including question answering, sentiment analysis, and named entity recognition. Pre-trained on a large corpus of text, BERT can be fine-tuned for specific tasks, demonstrating high versatility and effectiveness in diverse applications [32].

Recent research has explored various applications and enhancements of BERT in the domain of sentiment analysis. One study focused on aspect-based sentiment analysis (ABSA), leveraging BERT's contextual word representations and fine-tuning techniques to improve performance in out-of-domain tasks. The results demonstrated significant advancements over previous state-of-the-art models, highlighting BERT's robust performance in complex sentiment analysis scenarios [33]. Further investigations into BERT's efficacy for end-to-end aspect-based sentiment analysis (E2E-ABSA) have shown that even simple BERT-based architectures can outperform more complex models. By integrating BERT embeddings with neural networks, have consistently achieved superior results have consistently been achieved in capturing aspect-based sentiments and have demonstrated robustness against overfitting [34].

In addition to aspect-based sentiment analysis, BERT has been adapted for Review Reading Comprehension (RRC), transforming customer reviews into valuable sources of knowledge for answering user questions. A novel post-training approach on BERT has been proposed to enhance its performance in RRC and related sentiment analysis tasks, with experimental results showing significant improvements in understanding and extracting sentiments from reviews [35].

Moreover, BERT's combination with other neural network architectures, such as CNN, RNN, and BiLSTM, has proven to be highly effective for sentiment analysis on social media platforms like Twitter. These combinations have shown improved accuracy, precision, recall, and F1-score compared to traditional models, demonstrating BERT's ability to capture deeper contextual information and enhance sentiment analysis results [36]. When compared to unsupervised lexicon-based models, traditional supervised machine learning models, and advanced deep learning models, BERT consistently outperforms these techniques. This makes BERT a valuable tool for analytics professionals and researchers in various fields [37].

Finally, the transfer learning capabilities of BERT have been explored for sentiment analysis in multilingual contexts. By comparing BERT with other word embedding techniques like Word2Vec, GloVe, and fastText, it has been shown that BERT, when fine-tuned appropriately, significantly outperforms other models, demonstrating its potential in multilingual sentiment analysis [38].

2.5 VADER (Valence Aware Dictionary and Sentiment Reasoner)

VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. It uses a combination of qualitative and quantitative methods to provide a sentiment score for a given text. VADER is known for its simplicity and effectiveness in handling informal text, including slang, emoticons, and abbreviations, which are common in social media posts. The tool calculates a sentiment intensity score based on a dictionary of lexical features that are typically associated with positive, negative, and neutral sentiments [39].

Several studies have shown VADER's applicability across different domains. For instance, research on social media platforms such as TikTok and Twitter has shown that VADER can effectively classify sentiments in user-generated content. In studies involving sentiment analysis of TikTok reviews and tweets about cannabidiol (CBD), VADER was used to label sentiments, which were then analyzed to gain insights into user opinions and market trends. These studies highlighted VADER's capability to process large volumes of data and provide meaningful sentiment classifications, which are crucial for understanding public opinion [40, 41].

In the public health sector, VADER has been utilized to explore sentiments related to food security through the analysis of social media data. This approach has proven valuable in tracking public concerns and reactions to various issues, such as food scarcity and relief efforts. By analyzing sentiments over time, researchers can identify trends and shifts in public opinion, which can inform policy-making and public health interventions [42].

VADER's application in the financial sector has also been explored, particularly in predicting market volatility. Studies have found that sentiment analysis of financial news and tweets can correlate with market movements, providing a predictive signal for market trends [43].

Furthermore, VADER has been used in comprehensive evaluations to compare its performance with other sentiment analysis tools. Research has shown that VADER often outperforms traditional sentiment analysis methods, including machine learning-based approaches, in terms of accuracy and efficiency [44]. The latter make VADER an additional preferred choice for quick and reliable sentiment analysis in various contexts.

3 Proposed Methodology and Algorithms

3.1 Toy Example

This section presents a toy example to illustrate the problem of detecting dangerous products based on user reviews (Table 1) and the proposed solution using machine learning.

No.	Review Text	Product
1	As a concerned parent, I appreciate the strict safety regulations that Toy1 adheres to. It gives me peace of mind knowing my child is playing with a safe toy.	Toy1
2	Attention parents! Toy2 has been recalled due to a potential choking hazard. Stay informed and keep your children safe.	Toy2
3	Beware of Toy3's excessive noise levels. Prolonged exposure can harm children's hearing. Let's prioritize their well-being and demand quieter toys.	Toy3
4	Shocked to find out that Toy4 has small parts that pose a choking hazard. Let's hold toy companies accountable for ensuring child safety.	Toy4

Table 1: Users' Reviews Examples

The first diagram (Figure 1) illustrates the problem of a child playing with a toy that turns out to be dangerous. In this context, the "Girl" icon represents a child who interacts with a toy. The "Toy" icon stands for the product being used by the child, and the "Danger" icon signifies the potential risk or danger associated with the toy. Arrows between these icons indicate the actions involved: the child plays with the toy, and the toy includes some dangerous features. This visual representation helps to understand the real-world implications of identifying dangerous products.

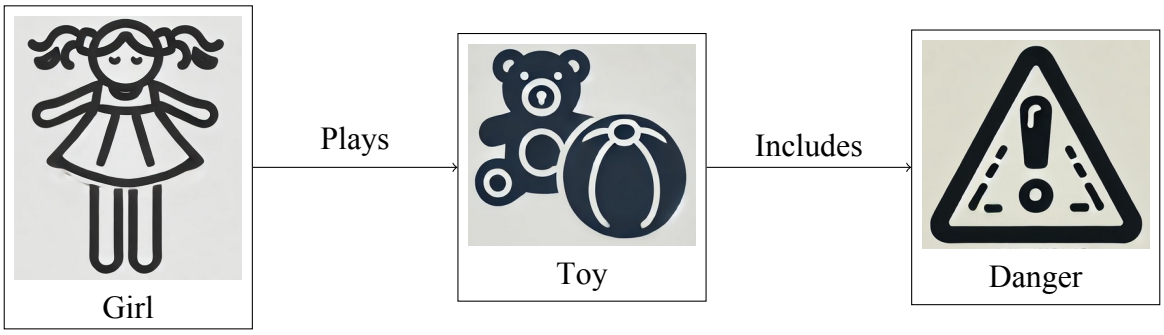


Figure 1: Problem to be addressed

The second diagram (Figure 2) demonstrates the detection process, which involves analyzing user reviews to assess the safety of a product. Here, the "User" icon depicts a person searching for information about a product. The "Toy" icon again represents the product under investigation, while the "User Reviews" icon symbolizes the feedback provided by users who have interacted with the product. The arrows illustrate the steps in this process: the user searches for the product, and the subsequent analysis of user reviews helps detect any potential dangers. This diagram shows how user feedback can be leveraged to evaluate product safety.

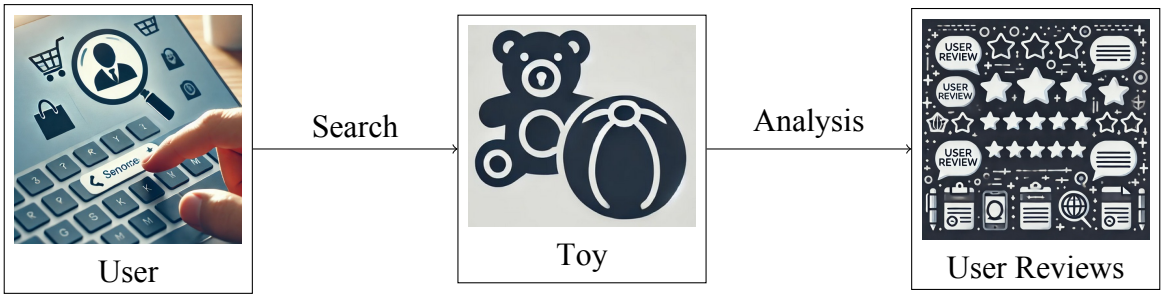


Figure 2: Detection of the problem

The third diagram (Figure 3) presents the proposed solution using machine learning. This involves training a machine learning model to classify products based on user reviews. In this scenario, the "User Reviews" icon represents the input data consisting of user feedback. The "ML Model" icon depicts the machine learning model that processes these reviews, and the "Conclusion" icon indicates the outcome of the analysis, specifying whether the product is dangerous or safe. The arrows demonstrate the flow of data: the machine learning model analyzes the user reviews and classifies the product accordingly. This diagram highlights the automated process of detecting dangerous products through advanced analysis techniques.

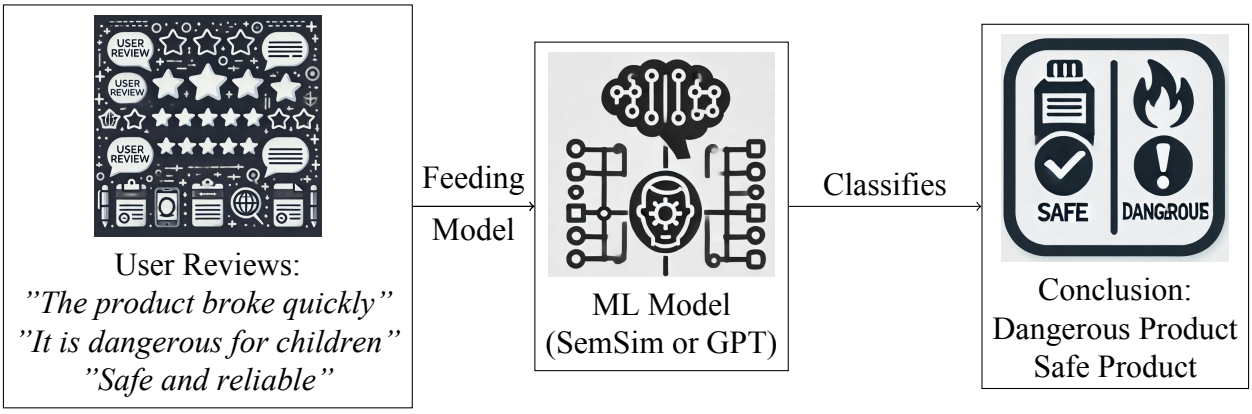


Figure 3: Solution

3.2 Results of a Toy example Using Wu-Palmer Semantic Similarity Measure

The algorithm initiates with the construction of an artificial list comprising terms associated with potential dangers, aligning with the EU's toy safety legislation. This lexicon forms the foundation for the subsequent analysis of user-generated content, specifically tweets, regarding various hypothetical products labeled as Toy 1, Toy 2, Toy 3, and Toy 4.

This toy example demonstrates the basic application of the algorithm, including preprocessing steps, similarity calculations, and the categorization of product reviews. Various visualizations are provided to illustrate the algorithm's functionality and the outcomes of the experiment.

Experimental Setup: The toy example involves a set of synthetic tweets, each containing information about different toys and potential safety concerns. The Wu-Palmer similarity measure is employed to compute the semantic similarity between terms in the tweets and a predefined list of dangerous terms. The tweets are then categorized based on their similarity scores.

Data Preprocessing: The tweets are preprocessed to extract significant attributes, remove stopwords, and convert terms to their base forms. This preprocessing step ensures that the similarity calculations are accurate and meaningful.

Calculation of Semantic Similarity: The Wu-Palmer similarity measure is used to calculate the semantic similarity between terms in the tweets and the dangerous terms. A weighted average of the similarity scores is computed for each tweet, which serves as the basis for categorizing the tweets.

Visualization of Results: To provide a comprehensive understanding of the algorithm's performance, several visualizations are generated:

First, we examine the relevant tweets extracted for the toy example (Table: 2). These tweets contain mentions of various toys and potential safety concerns. The extraction process highlights the tweets that are pertinent to the analysis, serving as the primary data for further evaluation.

Relevant Tweets
As a concerned parent, I appreciate the strict safety regulations that Toy1 adheres to. It gives me peace of mind knowing my child is playing with a safe toy. #Parenting #ToySafety
Parents bad quality and high risk! Be careful of Toy1's easily breakable parts. They can become sharp objects, endangering our children. Choose toys built to last and prioritize safety. #BreakageRisk #ChildSafety
Disappointed with the lack of durability in Toy1. It fell apart within days, and that's not what I expect from a toy marketed as safe. #QualityIssues #CustomerExperience
I regret purchasing Toy1. Its poor design and lack of safety features make it a potential danger to children. Spread the word to protect others! #UnsafeDesign #ToyFail
Warning: Toy1 poses a strangulation risk due to long cords. Parents, please be cautious and ensure your children's safety. #ChokingHazard #UnsafeToys
Just purchased Toy1 for my niece, and I'm relieved it meets all EU safety standards! #SafeToys #EURegulations
Kudos to Toy1 for its eco-friendly design and non-toxic materials. It's essential to prioritize both safety and sustainability. #GreenToys #SafePlay
Bad news! Very concerned to learn that Toy1 has paint containing high levels of lead toxicity. This is a clear violation of toy safety regulations. Let's hold manufacturers accountable! #LeadToxicity #ChildHealth

Table 2: Relevant tweets extracted for the toy example.

Next, we delve into the sentiment analysis of the relevant tweets. The tweets (Figure:4) are classified into positive and negative sentiments, providing insights into the overall tone and attitude of the user-generated content regarding the toys.

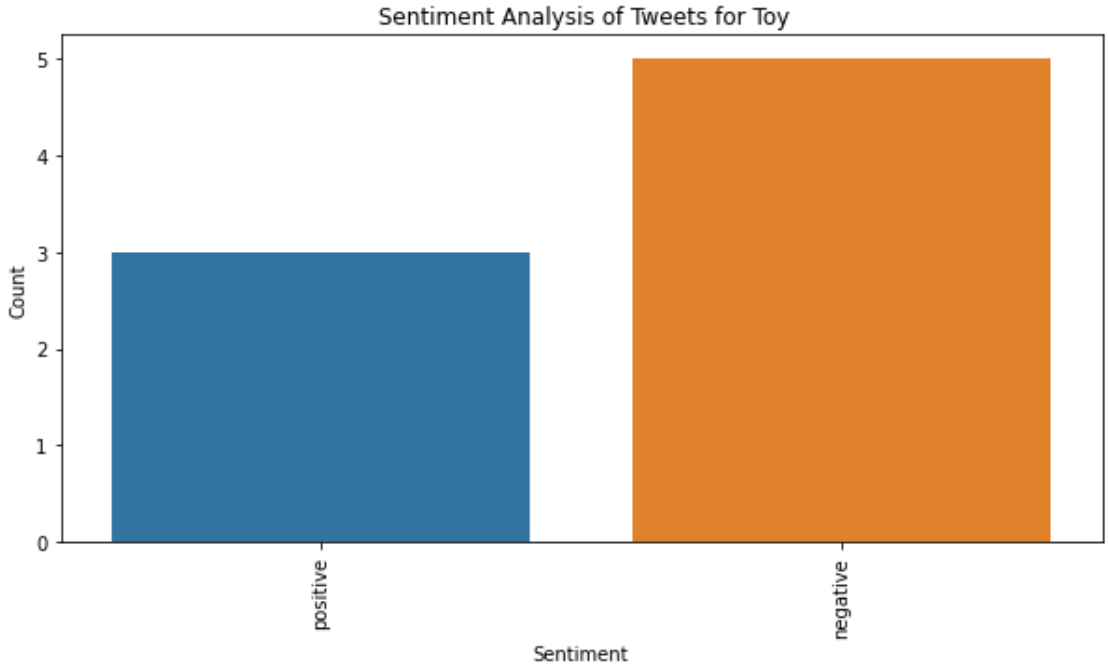


Figure 4: Sentiment analysis of tweets for a toy example.

A word cloud (Figure:5) presented below is generated in order to visualize the dangerous terms found in the related product's reviews. The size of each word reflects its frequency, providing a visual summary of the most common safety concerns mentioned in the reviews.

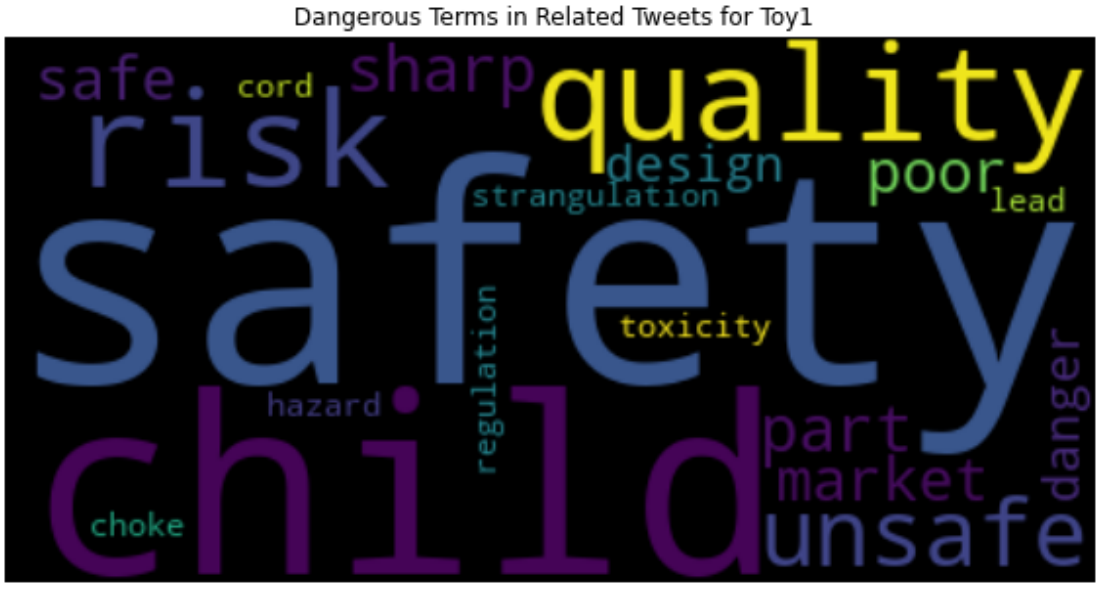
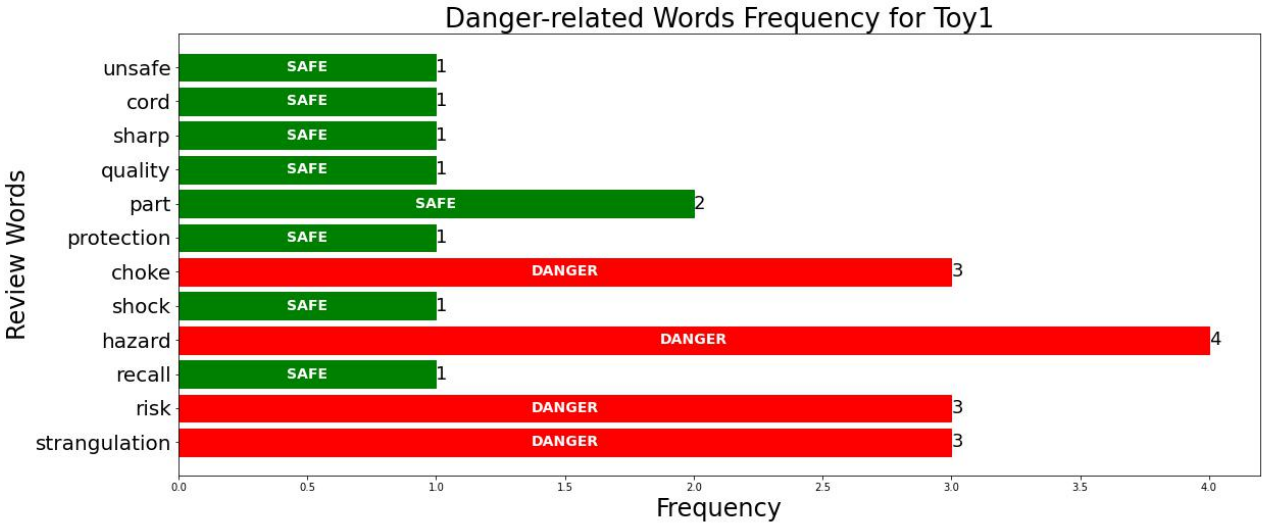


Figure 5: Wordcloud of dangerous terms found in related reviews for the toy example.

Moving forward, we observe the frequency of danger-related words associated with each toy attribute. This bar chart (Figure:6) highlights the prevalence of specific dangerous terms, helping to pinpoint the most frequently occurring safety issues.



1. Toy1 has received fantastic reviews for its high-quality construction and adherence to safety regulations. It's a must-have for any child's toy collection. #TopToy #CustomerFavorites

---

2. Toy1 passed rigorous safety tests with flying colors. It's a relief to know our little ones can enjoy hours of fun without any worries. #ChildSafety #QualityToys

---

3. Shocked to find out that Toy1 poses a strangulation and choking hazard. We must demand safer toys for our children. #ChildProtection #ToySafety

---

4. Beware of Toy1's strangulation risk. Long cords can easily choke children. Safety should always be the top priority. #StrangulationRisk #ChildSafety

---

5. Toy1 has been found to contain parts that can choke and strangle children. Parents, stay vigilant to ensure the safety of your little ones. #ChokingHazard #Strangulation

---

Figure 6: Danger-related words frequency for the toy example.

To further understand the data, we look at the semantic similarity values for each tweet (Figure:7) . A bar chart of the semantic similarity values for each tweet is presented. This chart shows the similarity scores, providing a comparative view of how each tweet aligns with the dangerous terms and how the 'safe' in green and the 'dangerous' in red tweets are categorized. The values are calculated using the Wu-Palmer measure, indicating the degree of similarity between the terms in each tweet and the dangerous terms list.

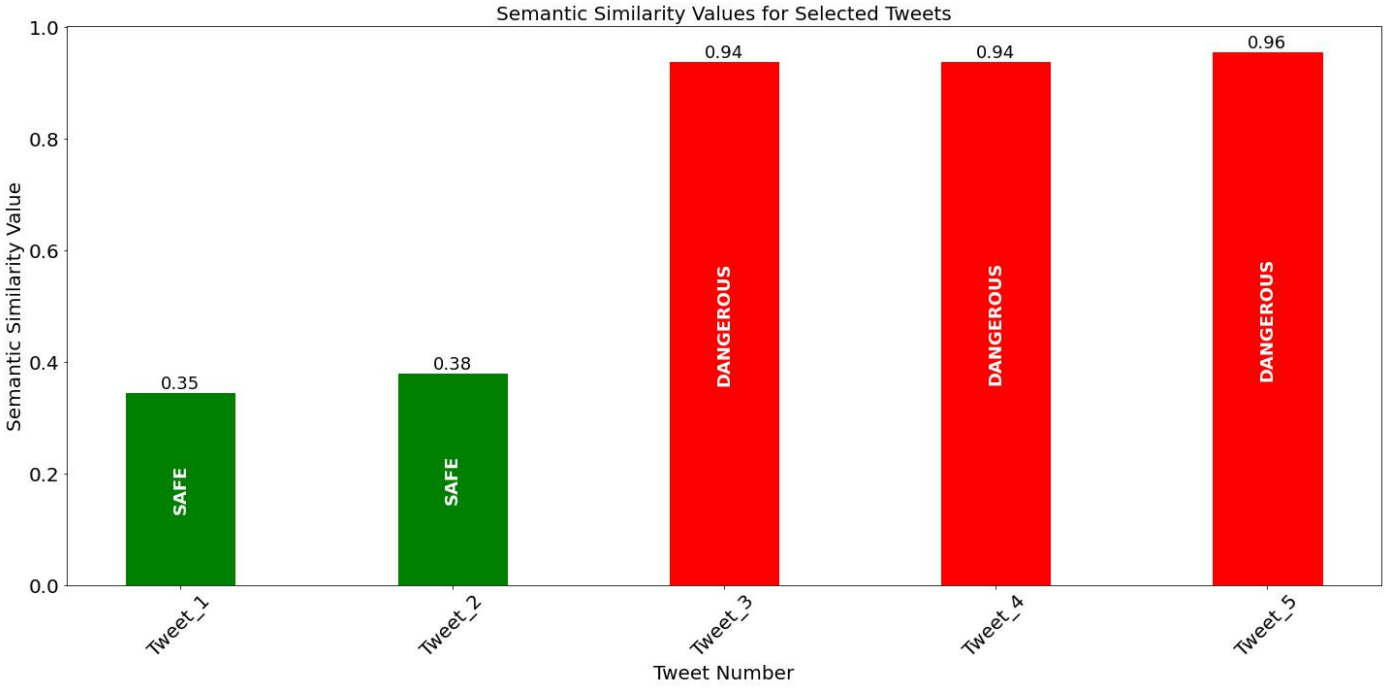


Figure 7: Bar chart of semantic similarity values for each tweet in the toy example using Wu-Palmer measure.

3.3 Methodology

The methodology presented here outlines the general approach used to evaluate product safety through user reviews. This approach is designed to be flexible and applicable across a range of experiments, each of which will be detailed in subsequent sections. The key stages of this methodology include data collection and pre-processing, semantic similarity calculation, analysis and visualization, and drawing conclusions based on the results.

- **Data Collection and Preprocessing:**

  The initial stage involves collecting user reviews relevant to the products under investigation. These reviews are then subjected to a comprehensive preprocessing phase to prepare the text data for analysis. This preprocessing includes steps such as tokenization, lemmatization, part-of-speech tagging, and the removal of stopwords, punctuation, and non-alphanumeric characters. By refining the text data to its most essential components, the preprocessing step ensures that the subsequent analysis is both accurate and efficient.

- **Semantic Similarity Calculation:**

  Central to this methodology is the calculation of semantic similarity, which quantifies the conceptual closeness between terms in the user reviews and a predefined danger lexicon. Various semantic similarity measures are employed to achieve this. The algorithm compares the lemmas extracted from the reviews against the terms in the lexicon to determine similarity scores, which are then weighted according to their relevance. This process allows for a nuanced understanding of how closely the content of the reviews aligns with known safety concerns.

- **Analysis and Visualization:**

  After calculating the semantic similarity scores, the methodology involves analyzing these scores to assess the safety of the products. This stage includes aggregating the scores to provide an overall safety assessment for each product. Visualization techniques are used to present the results of this analysis effectively. Graphs, plots, and word clouds are generated to illustrate the distribution of reviews, the outcomes of sentiment analysis, and the frequency of terms associated with potential dangers. These visualizations help in communicating the findings clearly and concisely.

- **Comparative Analysis:**

  To enhance the robustness of the safety evaluations, the methodology incorporates a comparative analysis using multiple semantic similarity measures. This multifaceted assessment allows for a more comprehensive evaluation of textual similarity. By comparing results from different measures, the methodology ensures that the safety assessments are reliable and not dependent on a single metric. This comparative approach provides a holistic view of potential safety concerns, making the evaluations more robust and credible.

- **Conclusion:**

  The methodology culminates in drawing conclusions based on the analyzed data. By systematically evaluating user reviews through sophisticated NLP and semantic analysis techniques, the methodology provides a detailed understanding of potential safety issues associated with products. These conclusions can help in identifying hazards and contribute to the development of safer products, benefiting both consumers and manufacturers. The approach ensures that safety evaluations are thorough and based on a robust analysis of user feedback.

3.4 Algorithm for Detecting if a Toy is Dangerous Based on Users' Reviews

The following diagrams provide a quick visual overview of the steps involved in the algorithm for detecting if a toy is dangerous based on users' reviews.

Figure 8: Filtering and Sentiment Analysis

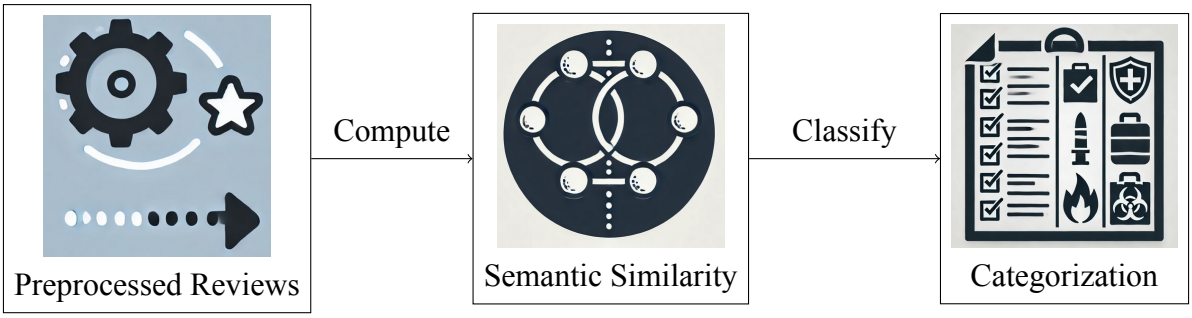


Figure 9: Semantic Similarity and Final Categorization

- **Input:**
  - D: List of risk-related terms (e.g., 'sodium', 'unsafe').
  - U: List of users' reviews referring to various toys.
- **Output:**
  - Categorization of toys as dangerous or safe based on the analysis of users' reviews.

  Symbols and Definitions:
  - U_{filtered}: Subset of reviews filtered to refer to a specific toy.
  - polarity(u): Sentiment polarity of review u.
  - preprocess(u): Preprocessing function applied to review u (lemmatization, stopword removal, etc.).
  - lemmas(u): Set of lemmas extracted from review u.
  - Sim$(w, d, \text{measure})$: Semantic similarity between lemma w and dangerous term d using the specified measure.
  - $w(l)$: Weight assigned to lemma l.
  - weighted_avg_sim(u): Weighted average similarity score for review u.
  - θ: Threshold for classifying a toy as dangerous or safe.
  - categorize_toy(u): Categorization of the toy as dangerous or safe based on the weighted average similarity score.
- **Procedure:**
  - **Step 1: Filter Reviews**
    - Select reviews that refer to a specific toy.

$$U_{\text{filtered}} = \{u \in U \mid \text{refers\_to\_toy}(u)\} \tag{8}$$

  - **Step 2: Sentiment Analysis**

- Classify reviews as positive or negative based on their polarity.

$$S(u) = \begin{cases} \text{positive} & \text{if polarity}(u) > 0 \\ \text{negative} & \text{if polarity}(u) < 0 \end{cases} \tag{9}$$

– **Step 3: Preprocessing**

- Lemmatization, removal of stopwords and symbols, and word standardization.

$$u_{\text{processed}} = \text{preprocess}(u), \quad \forall u \in U_{\text{filtered}} \tag{10}$$

– **Step 4: Semantic Similarity Calculation**

- Calculate the semantic similarity between each lemma in the review and the risk terms using various semantic similarity measures.

$$\text{Sim}(w, d, \text{measure}) = \max_{s_w \in S_w, s_d \in S_d} \text{sim}_{\text{measure}}(s_w, s_d) \tag{11}$$

- where w is the lemma from the review, d is each risk-related term, S_w and S_d are the synsets of lemmas w and terms d in WordNet, and $\text{sim}_{\text{measure}}(s_w, s_d)$ is the semantic similarity score using the specified measure.

– **Step 5: Weighting and Averaging**

- Apply weights to the semantic similarities and compute the weighted average similarity score.

$$\text{weighted\_avg\_sim}(u) = \frac{\sum_{l \in \text{lemmas}(u)} w(l) \cdot \text{sim}(l, D)}{\sum_{l \in \text{lemmas}(u)} w(l)} \tag{12}$$

– **Step 6: Final Categorization**

- Categorize the toy as dangerous if the average similarity score exceeds a threshold (θ).

$$\text{categorize\_toy}(u) = \begin{cases} \text{dangerous} & \text{if weighted\_avg\_sim}(u) > \theta \\ \text{safe} & \text{if weighted\_avg\_sim}(u) \leq \theta \end{cases} \tag{13}$$

3.4.1 Pseudocode

Algorithm 1 Simple Algorithm for Detecting if a Toy is dangerous

Input:
  D: List of risk-related terms (e.g., 'sodium', 'unsafe').
  U: List of users' reviews referring to various toys.
Output:
  Categorization of toys as dangerous or safe based on the analysis of users' reviews.
Procedure:
1. Filter Reviews:
  $U_{\text{filtered}} = \{u \in U \mid \text{refers\_to\_toy}(u)\}$
2. Sentiment Analysis:
for each u in U_{filtered} **do**
  if polarity(u) > 0 **then**
    sentiment(u) = positive
  else if polarity(u) < 0 **then**
    sentiment(u) = negative
  end if
end for
3. Preprocessing:
for each u in U_{filtered} **do**
  $u_{\text{processed}}$ = preprocess(u)
end for
4. Semantic Similarity Calculation:
for each u in U_{filtered} **do**
  for each lemma w in u **do**
    for each dangerous term d in D **do**
      Calculate semantic similarity $\text{Sim}(w, d, \text{measure})$
    end for
  end for
end for
5. Weighting and Averaging:
for each u in U_{filtered} **do**
  $\text{weighted\_avg\_sim}(u) = \frac{\sum_{l \in \text{lemmas}(u)} w(l) \cdot \text{sim}(l, D)}{\sum_{l \in \text{lemmas}(u)} w(l)}$
end for
6. Final Categorization:
for each u in U_{filtered} **do**
  if weighted_avg_sim(u) $>$ threshold **then**
    categorize_toy(u) = dangerous
  else
    categorize_toy(u) = safe
  end if
end for

3.5 Advanced Algorithm for Detecting if a Toy is Dangerous

In this advanced algorithm, several enhancements have been introduced to improve the detection of dangerous products based on user reviews. These enhancements include categorization of reviews, advanced preprocessing, vectorization of reviews, filtering of non-vocabulary words, creation of analysis terms, processing analysis terms with sentiment analysis, and normalization of similarity scores. These improvements aim to provide a more accurate and reliable categorization of products as dangerous or safe.

The following diagrams provide a quick visual overview of the new steps and improvements in the advanced algorithm compared to the initial algorithm, allowing for an easy understanding of the added enhancements.

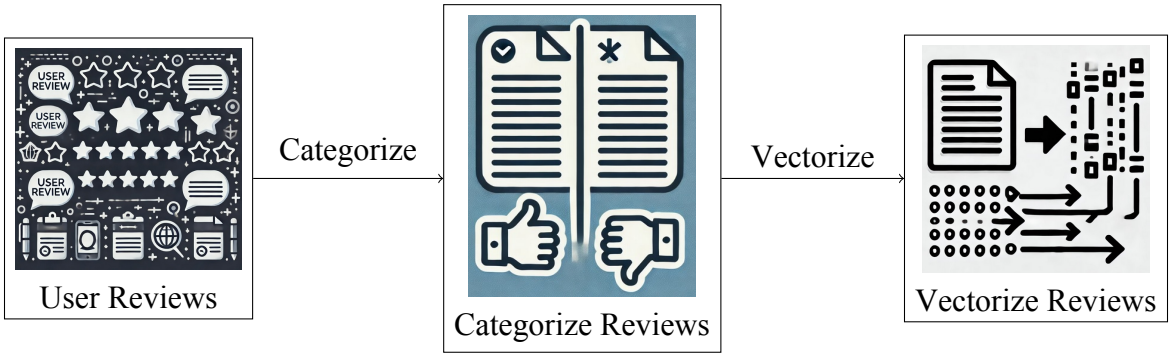


Figure 10: Categorization and Vectorization

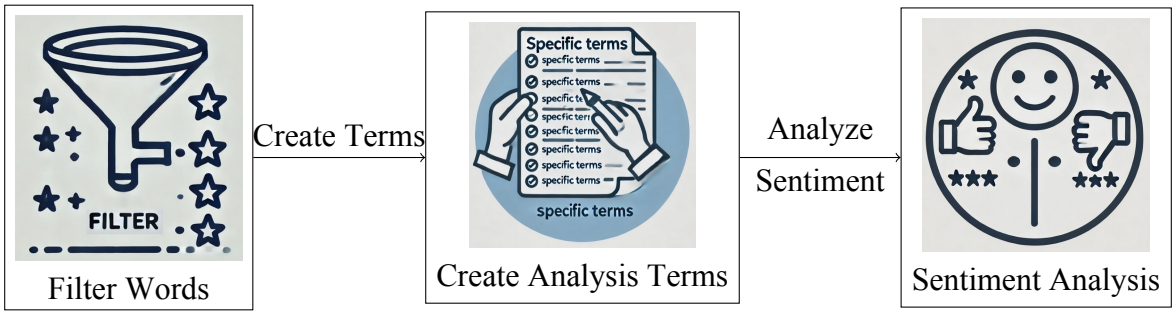


Figure 11: Filtering and Creating Analysis Terms

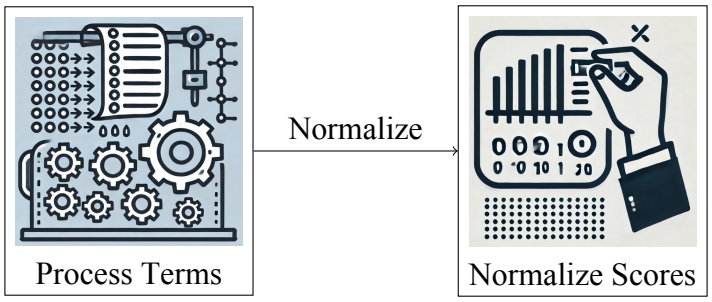


Figure 12: Processing and Normalizing

To enhance the detection of dangerous products based on user reviews, the advanced algorithm incorporates several improvements. These enhancements include more refined techniques for categorization, preprocessing, vectorization, and normalization, ensuring a more accurate and reliable analysis.

3.5.1 Symbols and Definitions

- $category(u)$: The category assigned to review u based on its rating.
- v_u: Vector representation of the processed review $u_{\text{processed}}$.
- $F(u_{\text{processed}})$: Set of words in the processed review $u_{\text{processed}}$ that are also in the vocabulary V.
- D: List of analysis terms.
- $D_{\text{sentiment}}$: Set of significant analysis terms based on sentiment analysis $S(d)$.
- Norm(X): Normalized value of the original score x in the set X.
- rating(u): Rating of the review u.
- u: Review.
- θ: Threshold rating above which the review is considered positive.
- U: Set of users' reviews referring to various products.
- U_{filtered}: Subset of reviews filtered to refer to a specific toy.
- preprocess(u): Preprocessing function applied to review u (lemmatization, stopword removal, etc.).
- $u_{\text{processed}}$: Processed review after preprocessing.
- V: Selected vocabulary.
- w: Word in the review.
- $S(d)$: Sentiment score of term d.
- x: Original score before normalization.
- similarity_scores: Set of similarity scores.
- $w(l)$: Weight assigned to lemma l.
- Sim$(w, d, \text{measure})$: Semantic similarity between lemma w and dangerous term d using the specified measure.
- weighted_avg_sim(u): Weighted average similarity score for review u.
- threshold: Threshold for classifying a product as dangerous or safe.
- categorize_toy(u): Categorization of the toy as dangerous or safe based on the weighted average similarity score.

3.5.2 Algorithm Description

- **Categorize Reviews:**
  - The reviews are categorized based on their rating to determine the ground truth of the reviews, classifying them as positive or negative. This step helps in assessing the sentiment of the reviews.

$$category(u) = \begin{cases} 1 & \text{if } rating(u) > \theta \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

    where u is the review, $rating(u)$ is the rating of the review, and θ is the threshold rating above which the review is considered positive.

- **Vectorization:**
  - Each preprocessed review is transformed into a vector representation, which is used for further analysis.

$$v_u = \text{Vectorize}(u_{\text{processed}}), \quad \forall u \in U \tag{15}$$

    where v_u is the vector representation of the processed review $u_{\text{processed}}$.

- **Filter Words Not in Vocabulary:**
  - Words that are not in the predefined vocabulary are filtered out to focus on relevant terms.

$$F(u_{\text{processed}}) = \{w \in u_{\text{processed}} \mid w \in V\} \tag{16}$$

    where $F(u_{\text{processed}})$ is the set of words in the processed review $u_{\text{processed}}$ that are also in the vocabulary V.

- **Create Analysis Terms List:**
  - A list of analysis terms is created either from user-provided terms or extracted from the text corpus.

$$D = \begin{cases} \text{user-provided terms} & \text{if provided} \\ \text{extract terms from text corpus} & \text{otherwise} \end{cases} \tag{17}$$

    where D is the list of analysis terms.

- **Process Analysis Terms with Sentiment Analysis:**
  - The analysis terms are processed with sentiment analysis to identify significant terms.

$$D_{\text{sentiment}} = \{d \in D \mid S(d) \text{ is significant}\} \tag{18}$$

    where $D_{\text{sentiment}}$ is the set of significant analysis terms based on sentiment analysis $S(d)$.

- **Normalize Similarity Scores:**

– The normalization is performed because not all semantic similarity measures are expressed on the same scale. By normalizing to the range [0,1], we ensure a consistent representation of the similarity scores.

$$\text{Norm}(X) = \frac{x - \min(X)}{\max(X) - \min(X)} \tag{19}$$

where $\text{Norm}(X)$ is the normalized value, x is the original score, and $\min(X)$, $\max(X)$ are the minimum and maximum scores in the set X.

3.5.3 Pseudocode

Algorithm 2 Advanced Algorithm for Detecting if a Product is Dangerous Based on User Reviews

Input:
  U: List of users' reviews referring to various products (e.g. toys).
  D: List of risk-related terms.
Output:
  Categorization of products as dangerous or safe based on the analysis of users' reviews.
Procedure:
1. Categorize Reviews:
for each review u in U **do**
  if rating(u) $> \theta$ **then**
    category(u) = 1
  else
    category(u) = 0
  end if
end for
2. Preprocessing:
for each review u in U **do**
  $u_{\text{processed}} = \text{preprocess}(u)$
end for
3. Vectorization:
for each preprocessed review $u_{\text{processed}}$ in U **do**
  $v_u = \text{Vectorize}(u_{\text{processed}})$
end for
4. Filter Words Not in Vocabulary:
for each preprocessed review $u_{\text{processed}}$ in U **do**
  $F(u_{\text{processed}}) = \{w \in u_{\text{processed}} \mid w \in V\}$
end for
5. Create Analysis Terms List:
if terms provided by user **then**
  $D =$ user-provided terms
else
  $D =$ extract terms from text corpus
end if
6. Process Analysis Terms with Sentiment Analysis:
  $D_{\text{sentiment}} = \{d \in D \mid S(d) \text{ is significant}\}$
7. Filter Reviews:
  $U_{\text{filtered}} = \{u \in U \mid \text{refers\_to\_product}(u)\}$

8. Calculate Sentiment Indicators:
for each review u in U_{filtered} **do**
    if S(u) $>$ 0 **then**
        sentiment(u) = 1
    else
        sentiment(u) = 0
    end if
end for
9. Semantic Similarity Calculation:
for each u in U_{filtered} **do**
    for each lemma w in u **do**
        for each term d in $D_{\text{sentiment}}$ **do**
            Calculate semantic similarity $\text{Sim}(w, d, \text{measure})$
        end for
    end for
end for
10. Normalize Similarity Scores:
for each score x in similarity_scores **do**
    $x_{\text{normalized}} = \frac{x - \min(\text{similarity\_scores})}{\max(\text{similarity\_scores}) - \min(\text{similarity\_scores})}$
end for
11. Weighting and Averaging:
for each u in U_{filtered} **do**

$$\text{weighted\_avg\_sim}(u) = \frac{\sum_{l \in \text{lemmas}(u)} w(l) \cdot \text{Sim}(l, D_{\text{sentiment}})}{\sum_{l \in \text{lemmas}(u)} w(l)}$$

end for
12. Final Categorization:
for each u in U_{filtered} **do**
    if weighted_avg_sim(u) $>$ threshold **then**
        categorize_toy(u) = dangerous
    else
        categorize_toy(u) = safe
    end if
end for

3.6 Ensemble Algorithm of different Semantics Similarity Measures (Boosting)

The Ensemble Algorithm of different Semantics Similarity Measures (Boosting) introduces an automated method for creating classes and calculating appropriate weights for semantic similarity (SemSim) scores. The primary goal is to ensure systematic and adaptive weighting of similarity scores based on their distribution. The algorithm normalizes the SemSim scores to a standard scale, divides them into quantile-based classes, and calculates the frequency of scores within each class. Exponential weights are then assigned based on these frequencies, with less frequent scores receiving higher weights. The weights are normalized to maintain proportional integrity. A threshold is determined based on class bounds, and the proportion of scores below this threshold is used to adjust the weighting parameter α. Finally, the weights are fine-tuned based on the overall sentiment of the reviews, ensuring that the weighting scheme accurately reflects the score distribution and sentiment, leading to more accurate and reliable categorization of products based on user reviews.

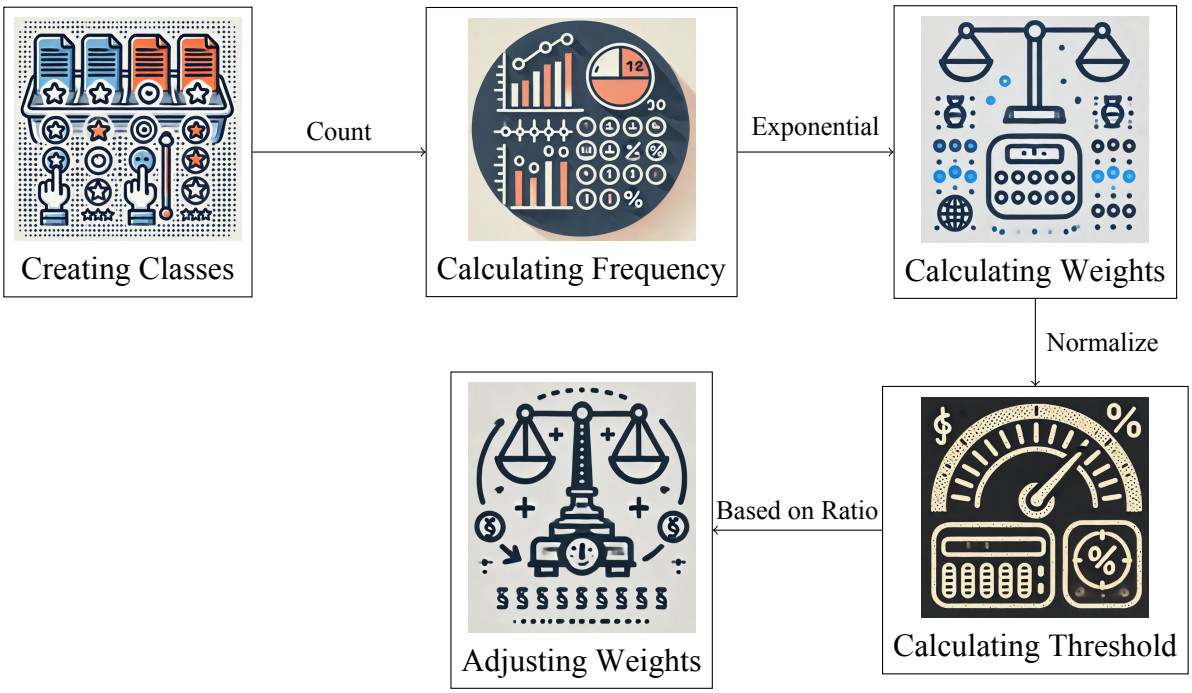


Figure 13: Visual Representation of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting)

Below a detailed explanation of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) is provided, focusing on the automated creation of classes and the calculation of weights for semantic similarity scores. This approach enhances the weighting mechanism by ensuring that weights are adaptively assigned based on score distribution and sentiment analysis.

3.6.1 Symbols and Definitions

- Score: Similarity score.
- Min Score: Minimum similarity score.
- Max Score: Maximum similarity score.

- Normalized Score: Normalized similarity score.
- Classes: Partitioned score classes.
- class_count$_i$: Number of items in the i-th class.
- c: Class for which the weight W is calculated.
- a: Parameter that adjusts the sensitivity of the weights to the distribution of the classes.
- total_count: Total number of items.
- $W(c)$: Weight for class c.
- Threshold: Threshold for fair weight distribution.
- below_threshold_count: Number of scores below the threshold.
- total_count: Total number of scores.
- ratio: Proportion of scores below the threshold.
- α: Adjusted parameter for the exponential weight function.
- base_alpha: Base value for α.

3.6.2 Algorithm Description

Step 1: Creation of SemSim Score Classes

The normalized SemSim scores are partitioned into quantile-based classes to categorize the distribution of similarity scores.

$$\text{Classes} = \text{quantile}(\text{SemSim Scores}, [0.2, 0.4, 0.6, 0.8, 1.0]) \tag{20}$$

This step ensures that the scores are divided into meaningful segments, which will be used for calculating weights.

Step 2: Calculation of Frequency for Each Class

In this step, the distribution of semantic similarity scores across the defined classes is determined. By counting the number of scores that fall into each class, we can analyze the frequency and density of scores within each range. This information is crucial for calculating the weights in the next step, as it reflects the relative importance and representation of each class in the overall dataset.

$$\text{class\_count}(c) = \sum_{s \in S} I(s \in c) \tag{21}$$

where: - class_count(c) is the number of scores in class c, - S is the set of normalized scores, - $I(s \in c)$ is an indicator function that equals 1 if score s falls into class c and 0 otherwise.

Step 3: Calculation of Exponential Weights

The weight for each class is calculated using an exponential function based on the frequency of scores within the class. This method assigns higher weights to less frequent classes, emphasizing rarer but potentially more significant scores.

$$W(c) = e^{-a \cdot \left(\frac{\text{class\_count}_i}{\text{total\_count}}\right)} \tag{22}$$

where: - class_count$_i$ is the number of items in the i-th class. - c is the class for which the weight W is calculated. - a is a parameter that adjusts the sensitivity of the weights to the distribution of the classes. - total_count is the total number of items.

Step 4: Normalization of Weights

To ensure the weights are proportional and their sum equals 1, they are normalized:

$$W(c) = \frac{W(c)}{\sum_{c \in \text{Classes}} W(c)} \tag{23}$$

where $\sum_{c \in \text{Classes}} W(c)$ is the sum of all weights.

Step 5: Calculation of Threshold

A threshold is determined based on the upper bound of the last class with a maximum value less than or equal to 0.5. This threshold helps in distinguishing significant scores from less significant ones.

$$\text{Threshold} = \max(\text{upper bound of the last class with max} \leq 0.5) \tag{24}$$

Step 6: Calculation of the Proportion of SemSim Scores Below the Threshold

The proportion of scores below the calculated threshold is computed. This ratio is crucial for adjusting the parameter a in the subsequent step.

$$\text{ratio} = \frac{\text{below\_threshold\_count}}{\text{total\_count}} \tag{25}$$

where: - below_threshold_count is the number of scores below the threshold. - total_count is the total number of scores.

Step 7: Adjusting the Parameter a Based on the Ratio

The parameter a is adjusted according to the ratio of scores below the threshold. If the ratio is less than 0.5, indicating fewer scores below the threshold, a is increased to give higher weights to more significant scores.

$$a = \text{base\_alpha} + (0.5 - \text{ratio}) \tag{26}$$

where: - base_alpha is the base value for a, usually initialized to 1.0. - $0.5 -$ ratio is the adjustment based on the observed ratio of low scores.

Step 8: Adjusting Weights Based on the Sentiment of Reviews

The weights are further adjusted based on the sentiment of the reviews to ensure that the weighting reflects the overall sentiment correctly. If all reviews are positive, the weights of classes with low scores are reduced. Conversely, if all reviews are negative, the weights of classes with high scores are reduced.

$$W(c) = \begin{cases} W(c_{\text{low}}) \times 0.1 & \text{if all reviews are positive and } c \text{ is a low score class} \\ W(c_{\text{high}}) \times 0.1 & \text{if all reviews are negative and } c \text{ is a high score class} \end{cases} \tag{27}$$

where: - $W(c_{\text{low}})$ is the weight for classes with low scores. - $W(c_{\text{high}})$ is the weight for classes with high scores.

3.6.3 Pseudocode for Algorithm 3

Algorithm 3 Ensemble Algorithm of different Semantics Similarity Measures (Boosting)

Input:
  S: List of semantic similarity scores.
Output:
  Classes and weights.
Procedure:
Step 1: Create Score Classes
Classes = quantile(S, [0.2, 0.4, 0.6, 0.8, 1.0])
Step 2: Calculate Frequency for Each Class
for each class c in Classes **do**
    class_count[c] = count(scores in c)
end for
Step 3: Calculate Exponential Weights
for each class c in Classes **do**
    $W(c) = e^{-\alpha \cdot \left(\frac{\text{class\_count}[c]}{\text{total\_count}}\right)}$
end for
Step 4: Normalize Weights
sum_weights = sum($W(c)$ for all c in Classes)
for each class c in Classes **do**
    $W(c) = \frac{W(c)}{\text{sum\_weights}}$
end for
Step 5: Calculate Threshold
threshold = max(upper bound of the last class with max ≤ 0.5)
Step 6: Calculate the Proportion of Scores Below the Threshold
below_threshold_count = count(scores $\leq$ threshold)
total_count = count(S)
ratio = below_threshold_count / total_count
Step 7: Adjust the Parameter a Based on the Ratio
$\alpha =$ base_alpha $+ (0.5 -$ ratio$)$
Step 8: Adjust Weights Based on the Sentiment of Reviews
if all reviews are positive **then**
    for each class c with low scores **do**
        $W(c) = W(c) \times 0.1$
    end for
else if all reviews are negative **then**
    for each class c with high scores **do**
        $W(c) = W(c) \times 0.1$
    end for
end if

4 Experimental Results

This chapter explores the practical aspects of the research by detailing the experimental setup and presenting the results obtained from testing the proposed algorithms on various datasets. It is structured in such a way in order to provide a comprehensive understanding of how the methodologies and algorithms introduced earlier are applied in practice, showcasing their performance and effectiveness.

It begins with a thorough explanation of the experimental protocol, outlining the steps taken to ensure a rigorous and systematic evaluation of the algorithms. This section covers the design of the experiments, the selection of datasets, and the metrics used to assess performance. The datasets include both synthetic data, created to simulate various scenarios, and real-world data from Amazon product reviews, providing a diverse range of test cases for the algorithms.

Following the protocol, the results of the experiments are presented, where each algorithm is evaluated based on its ability to detect dangerous products from user reviews. This section includes detailed analyses, supported by graphs and tables, to illustrate the performance of each algorithm under different conditions. The results highlight the strengths and weaknesses of the proposed methodologies, offering insights into their practical applicability.

A key focus of this chapter is the comparison of semantic similarity measures. The analysis shows how different measures impact the performance of the algorithms, providing a nuanced understanding of which measures are most effective in various contexts. This comparative analysis is crucial for identifying the best approaches to semantic similarity in the task of product safety assessment.

The chapter also includes an evaluation of Large Language Models (LLMs), exploring the application of advanced machine learning techniques to enhance the accuracy of the algorithms. This section examines how LLMs, such as those incorporating Retrieval-Augmented Generation (RAG), contribute to the overall performance, demonstrating the potential of these models to improve semantic analysis and classification tasks.

Finally, the limitations and potential improvements identified through the experimental results are discussed. This section offers a critical reflection on the findings, suggesting areas for further research and development to enhance the algorithms' robustness and reliability.

Through this chapter, readers will gain a comprehensive understanding of the practical implementation and performance of the proposed methodologies, providing a solid foundation for further advancements in the field of product safety assessment using semantic analysis and machine learning.

4.1 Experimental Protocol

The experimental protocol outlines the experimental procedure followed to evaluate the classification algorithms for detecting dangerous products based on user reviews. The aim is to clearly present the steps taken to ensure a rigorous and systematic assessment.

Experimental Design: The experimental design included specifying the algorithms to be designed and evaluated, the datasets to be used, and the performance metrics. The created algorithms encompass the aforemen-

tioned, in the previous chapters, semantic similarity computation measures, as well as the GPT-3.5 turbo, the Large Language Model (LLM) from OpenAI.

Review Classification Process: The process followed by the algorithm to classify reviews as dangerous or safe includes the following steps:

1. **Data Collection and Preprocessing:** Reviews are collected and preprocessed to remove stopwords, perform lemmatization, and extract significant terms.
2. **Semantic Similarity Calculation:** Each review is analyzed to calculate the semantic similarity between its terms and a predefined lexicon of dangerous terms.
3. **Scoring and Categorization:** Based on the semantic similarity scores, each review is classified as dangerous or safe. The algorithms are evaluated based on accuracy, recall, and other performance metrics.

Performance Metrics: The primary metrics used to evaluate the algorithms' performance include precision, recall, F1-score, and ROC curve. These metrics provide a comprehensive view of the algorithms' effectiveness in distinguishing between dangerous and safe products.

Through this process, the algorithms are evaluated consistently and accurately, allowing for objective comparison and identification of the best approaches for analyzing product reviews.

4.2 Datasets

In this section, the datasets used for the experimental evaluation of the algorithms are described. The experiments utilized two main datasets: synthetic data and real-world data from Amazon product reviews. The synthetic data was created to simulate various scenarios, while the Amazon data provides real product reviews, ensuring a diverse range of test cases for the algorithms.

4.2.1 Synthetic Dataset

The synthetic dataset is a collection of artificially generated tweets designed to simulate user reviews for products. It was developed to contain reviews based on the categories of product safety according to EU legislation. This artificial dataset aims to emulate real-world scenarios, where users express their concerns and experiences regarding the safety and quality of products. Below is an example of the synthetic dataset used in the experiments:

Example dataset of tweets:
"As a concerned parent, I appreciate the strict safety regulations that Toy1 adheres to. It gives me peace of mind knowing my child is playing with a safe toy. #Parenting #ToySafety"
"Attention parents! Toy2 has been recalled due to a potential choking hazard. Stay informed and keep your children safe. #ToyRecall #SafetyFirst"
"Parents, beware of Toy2! Recent reports suggest it may contain lead paint, posing serious health risks. Let's demand safer options for our kids. #LeadFreeToys #ParentingAlert"
"Beware of Toy3's excessive noise levels. Prolonged exposure can harm children's hearing. Let's prioritize their well-being and demand quieter toys. #NoiseHazard #ChildrensHealth"
"Shocked to find out that Toy4 has small parts that pose a choking hazard. Let's hold toy companies accountable for ensuring child safety. #ChildProtection #ToyManufacturers"
"Parents bad quality and high risk! Be careful of Toy1's easily breakable parts. They can become sharp objects, endangering our children. Choose toys built to last and prioritize safety. #BreakageRisk #ChildSafety"
"Shocked to discover that Toy2 contains small parts that can easily be swallowed. This violates EU safety regulations. Protect our children! #ChokingRisk #ChildSafety"
"Toy3's small magnets are a serious ingestion hazard. Manufacturers must ensure secure closures to prevent life-threatening accidents. #MagnetIngestion #ChildProtection"
"Disappointed with the lack of durability in Toy1. It fell apart within days, and that's not what I expect from a toy marketed as safe. #QualityIssues #CustomerExperience"
"Toy2's sharp edges are a serious concern. Children's toys should be designed with rounded edges to prevent injuries. #ChildSafety #DesignFlaws"

Table 3: Example of the syntheric dataset of tweets simulating product reviews and safety concerns.

This dataset (available a sample in the Appendix) simulates various safety issues and quality concerns that users might express in their reviews. It includes categories such as choking hazards, chemical safety, fire hazards, and durability, reflecting the broad range of safety considerations addressed by EU regulations.

To enhance the classification and analysis of these reviews, a comprehensive list of terms (Table:4) associated with danger, as defined by EU legislation, was created. This list functions as a thesaurus or a bag of related words, capturing the breadth of vocabulary used to describe various hazards and safety issues. This helps in accurately identifying and categorizing the potential risks mentioned in the reviews. The following is an excerpt of this list:

Table 4: Danger Related Terms

Chemical Hazards	**Mechanical Hazards**	**Choking Hazards**
sodium	insubstantial	small parts
toxicity hazards	mechanical hazards	swallow
allergic reaction	pinch points	ingestion hazard
mutagenic substances	sharp edges	choking
lead	inferior	suffocation
ammonia	projectiles	inhalation
radiation	fall hazards	aspiration
endocrine disruptors	entanglement hazard	cord
cadmium	heat hazard	button
mercury	pinch	magnet ingestion
arsenic	cutting	projectile
phthalates	laceration hazard	
formaldehyde	improper	
toxic materials	faulty design	
Fire and Explosion Hazards	**Electrical Hazards**	**General Safety Hazards**
flammable	electrocution	unsafe
explosion hazard	electrical hazard	risk
burn	electric shock	danger
fire hazard	CE marking	unsafe
inflammability		poor design
explosive		non-compliant
radiological hazards		

This comprehensive list supports the classification and analysis tasks by providing a robust vocabulary of terms associated with various dangers and hazards. By leveraging this thesaurus, the system can more effectively identify and categorize potential safety issues mentioned in user reviews, aligning with the EU's stringent safety standards for consumer products. The complete synthetic dataset is available on GitHub at the following link: `https://lmy.de/jjdwj`

4.2.2 Amazon Reviews Dataset

The Amazon reviews dataset [45] used in this study is a subset specifically focusing on the "Toys and Games" category. This dataset is a comprehensive collection of reviews from the Amazon platform, which includes not only ratings and text reviews but also product metadata such as descriptions, category information, price, brand, and image features. This subset was chosen due to its relevance to product safety and user experience, particularly in the context of toy safety and quality.

This updated version of the Amazon review dataset, initially released in 2014, encompasses a vast number of reviews, providing a rich source of data for analysis. Key features of the dataset include:

- **More reviews:** The dataset contains 233.1 million reviews, significantly more than the 142.8 million reviews in the 2014 version.

- **Newer reviews:** The reviews span from May 1996 to October 2018, offering a broad temporal range.
- **Detailed metadata:** Each review includes transaction metadata, product information (e.g., color, size, package type), product images, detailed product descriptions, technical details, and similar products.
- **Expanded categories:** The dataset includes reviews across multiple product categories, with five new categories added in this version.

In particular, the "Toys and Games" category, which is the focus of our analysis, consists of 1,828,971 reviews and 8,201,231 ratings. This subset is crucial for studying any issues raised by users in their reviews of toys and games.

To give a clearer picture of the dataset, here is an example of the data format used in this study:

4.2.3 Amazon Reviews Dataset

The Amazon reviews dataset [45] used in this study is a subset specifically focusing on the "Toys and Games" category. This dataset is a comprehensive collection of reviews from the Amazon platform, which includes not only ratings and text reviews but also product metadata such as descriptions, category information, price, brand, and image features. This subset was chosen due to its relevance to product safety and user experience, particularly in the context of toy safety and quality.

This updated version of the Amazon review dataset, initially released in 2014, encompasses a vast number of reviews, providing a rich source of data for analysis. Key features of the dataset include:

- **More reviews:** The dataset contains 233.1 million reviews, significantly more than the 142.8 million reviews in the 2014 version.
- **Newer reviews:** The reviews span from May 1996 to October 2018, offering a broad temporal range.
- **Detailed metadata:** Each review includes transaction metadata, product information (e.g., color, size, package type), product images, detailed product descriptions, technical details, and similar products.
- **Expanded categories:** The dataset includes reviews across multiple product categories, with five new categories added in this version.

In particular, the "Toys and Games" category, which is the focus of our analysis, consists of 1,828,971 reviews and 8,201,231 ratings. This subset is crucial for studying any issues raised by users in their reviews of toys and games.

4.3 Evaluation Metrics

The assessment of an algorithm's effectiveness, particularly in the context of classification tasks, hinges on the application of various evaluation metrics. These metrics offer quantitative insights that reflect the performance

of the algorithm across different dimensions, from its accuracy in making predictions to its precision, recall, and ability to discriminate between classes. This chapter delves into the core evaluation metrics used in the realm of machine learning and data analysis, providing a foundation for understanding the efficacy of the semantic similarity-based classification algorithm presented in this report.

In the following subsections, each metric is explored in detail, providing formulas, interpretations, and illustrative diagrams to enhance understanding. These metrics collectively form a robust framework for evaluating the performance of classification algorithms, offering insights that guide improvements and refinements.

4.3.1 Accuracy

Accuracy is one of the most straightforward and comprehensible evaluation metrics in classification problems. It provides a general idea of how well the model correctly predicts the categories, regardless of the class.

The formula for accuracy is as follows:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Instances}} \tag{28}$$

In this equation:

- True Positives (TP): The number of cases that the model correctly predicted the positive class.
- True Negatives (TN): The number of cases that the model correctly predicted the negative class.
- Total Instances: The total number of cases in the dataset.

Example

Consider a confusion matrix for a classification system with 100 cases, where:

- True Positives = 50
- True Negatives = 30
- False Positives = 10 (Positive cases that were incorrectly predicted as negative)
- False Negatives = 10 (Negative cases that were incorrectly predicted as positive)

The accuracy of the model is calculated as:

$$\text{Accuracy} = \frac{50 + 30}{100} = 0.8 \tag{29}$$

This means that the model correctly predicted 80% of the cases. Accuracy is a useful metric when the datasets are relatively balanced between classes, but it can be misleading in cases of class imbalance.

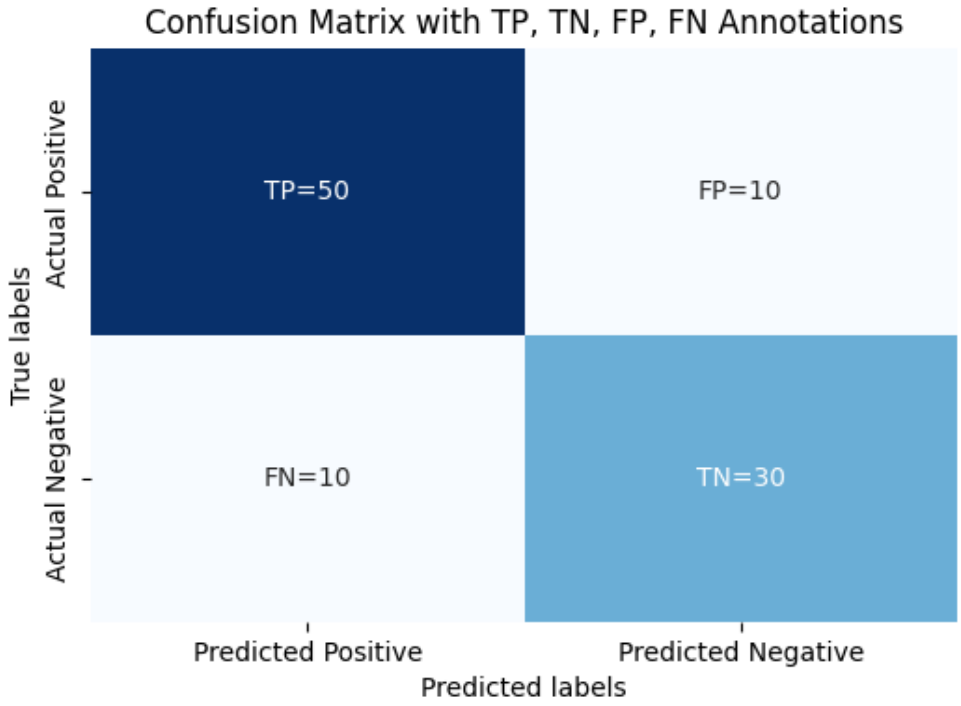


Figure 14: Confusion matrix illustrating the calculation of accuracy with True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

4.3.2 Precision

Precision, also known as the Positive Predictive Value, is a crucial metric in classification problems, especially when the costs of False Positives are high. It measures the proportion of correctly identified positive cases from all predicted positive cases. In simpler terms, it answers the question: "Of all the instances we labeled as positive, how many actually were positive?"

The formula for Precision is given by:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{30}$$

In this formula: - **True Positives (TP)** are the instances correctly identified as positive, - **False Positives (FP)** are the instances incorrectly labeled as positive, which are actually negative.

Example:

Consider a scenario in a medical diagnosis where a test predicts whether a tumor is malignant (positive class) or benign (negative class). Let's assume:

- True Positives (TP) = 40 (cases correctly identified as malignant),
- True Negatives (TN) = 50 (cases correctly identified as benign),

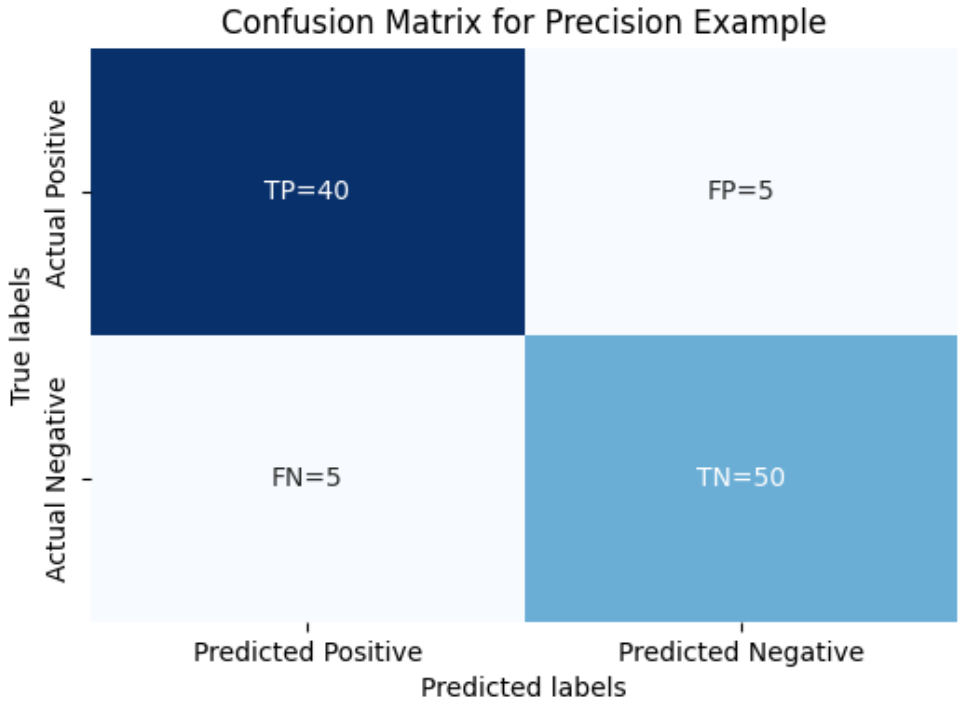


Figure 15: Confusion matrix illustrating the calculation of precision with True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

- False Positives (FP) = 5 (benign cases incorrectly identified as malignant),
- False Negatives (FN) = 5 (malignant cases incorrectly identified as benign).

With these values, the Precision of the model can be calculated as:

$$\text{Precision} = \frac{40}{40 + 5} = 0.89 \tag{31}$$

This result indicates that 89% of the instances that the model identified as malignant were actually malignant. Precision is particularly useful in contexts where the cost of False Positives is significant. For instance, in email spam detection, a False Positive (labeling a legitimate email as spam) could mean missing an important email, so a high Precision model would be desirable to minimize this risk.

4.3.3 Recall

Recall, also known as Sensitivity or True Positive Rate, is a critical metric in the field of machine learning and statistics, particularly within classification problems. It measures the model's ability to correctly identify all relevant instances within a dataset. In other words, Recall assesses how good a model is at capturing and correctly classifying the positive class instances out of all actual positive instances available.

The formula for Recall is expressed as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{32}$$

In this equation:

- **True Positives (TP)** refers to the instances that were correctly predicted as positive by the model.
- **False Negatives (FN)** are the instances that belong to the positive class but were incorrectly predicted as negative by the model.

Example:

Imagine a medical screening scenario where a model is used to predict the presence of a disease in patients. Let's say the model is tested on 100 patients:

- True Positives = 30: Patients correctly identified as having the disease.
- False Negatives = 20: Patients who have the disease but were missed by the model.
- False Positives (FP) = 10: Healthy patients incorrectly identified as having the disease.
- True Negatives (TN) = 40: Healthy patients correctly identified.

Given these values, the Recall of the model would be calculated as:

$$\text{Recall} = \frac{30}{30 + 20} = 0.6 \tag{33}$$

This result implies that out of all the actual disease cases, the model correctly identified 60% of them. Recall is particularly important in situations where missing a positive instance carries a significant cost, such as in disease detection or fraud prevention scenarios. It tells us how effective the model is in capturing all the relevant cases without necessarily focusing on the correctness of the negative predictions.

Confusion Matrix for Recall:

4.3.4 ROC and AUC

The Receiver Operating Characteristic (ROC) curve is a graphical representation of the performance of a classification model at all classification thresholds. It plots the True Positive Rate (Recall) against the False Positive Rate (FPR), which is calculated as:

$$\text{False Positive Rate (FPR)} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}} \tag{34}$$

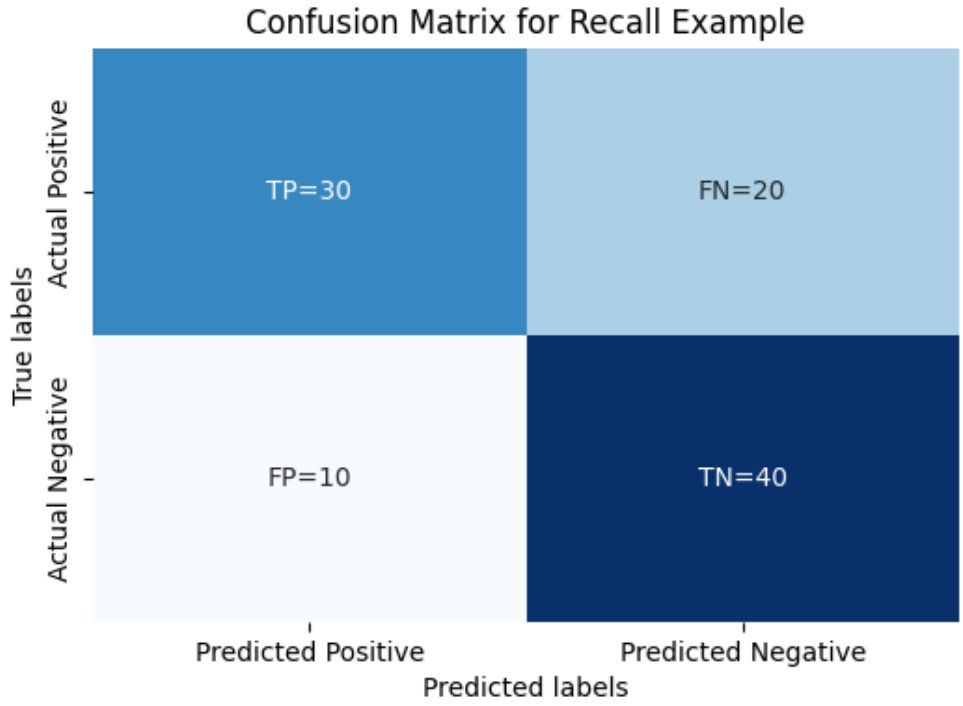


Figure 16: Confusion matrix highlighting the components relevant for calculating Recall.

The Area Under the ROC Curve (AUC) provides an aggregate measure of performance across all possible classification thresholds. The AUC value ranges from 0 to 1, with a higher value indicating better model performance. An AUC of 0.5 suggests no discriminative power, equivalent to random guessing, whereas an AUC of 1.0 indicates perfect discrimination between classes.

Example:

Consider the following values from a binary classification task:

- True Positives (TP) = 80
- False Positives (FP) = 20
- True Negatives (TN) = 90
- False Negatives (FN) = 10

The True Positive Rate (Recall) and False Positive Rate (FPR) are calculated as:

$$\text{True Positive Rate} = \frac{80}{80 + 10} = 0.89 \tag{35}$$

$$\text{False Positive Rate} = \frac{20}{20 + 90} = 0.18 \tag{36}$$

Plotting these rates at various threshold levels generates the ROC curve, and the AUC provides a single scalar value summarizing the model's ability to discriminate between positive and negative classes.

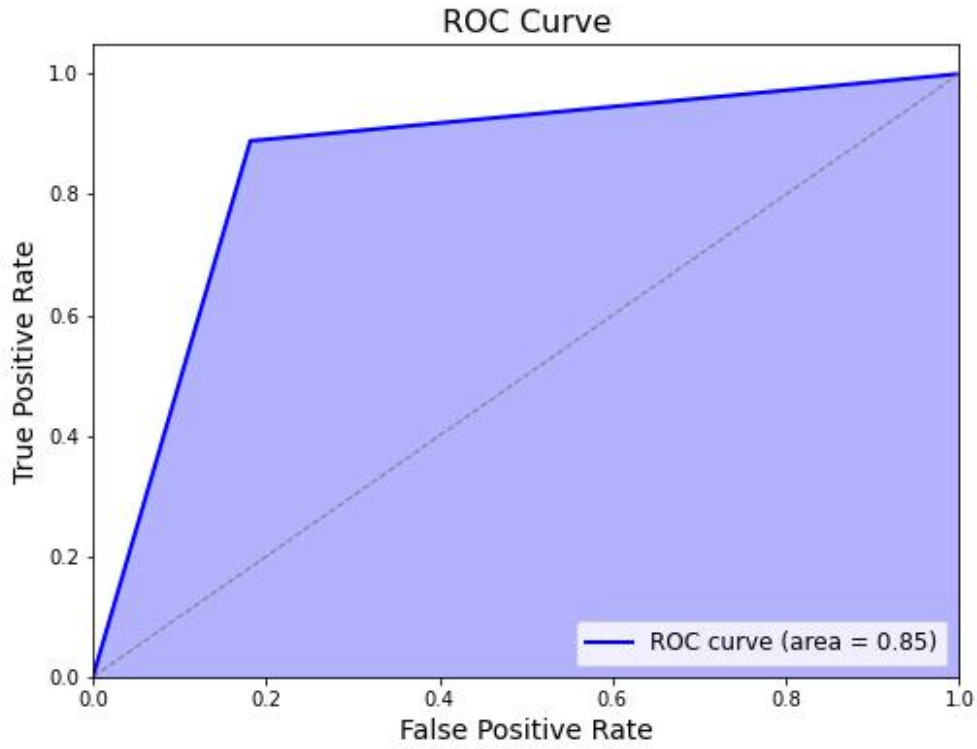


Figure 17: ROC curve illustrating the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR) for different threshold levels.

The ROC curve and AUC are particularly useful in evaluating the performance of models when dealing with imbalanced datasets, providing a more comprehensive understanding of the trade-offs between sensitivity (recall) and specificity across different decision thresholds.

4.4 Experiment 1: Results of the Main Algorithm Using Various Semantic Similarity Measures

The following section presents the results obtained after applying the main algorithm in both the synthetic and Amazon dataset. While introductory, these results give a good view of what the potential of the algorithm developed is.

Precision-Recall (PR) curves are critical tools for assessing the trade-off between precision and recall, especially in datasets with a significant imbalance between classes. The PR curve plots the precision (y-axis) against the recall (x-axis) for different threshold values, providing insight into the model's ability to retrieve relevant instances among all relevant instances it tries to retrieve. Interpolation of these curves allows for a smoother representation, highlighting the model's performance across different levels of recall. This graphical representation is particularly useful for identifying thresholds where precision significantly drops or remains stable, offering a visual means to pinpoint an optimal balance for decision-making processes in classification tasks.

Additionally, the ROC (Receiver Operating Characteristic) curve is a graphical representation that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. It is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings. The Area Under the Curve (AUC) provides a single scalar value that summarizes the overall performance of the classification system.

4.4.1 Application on Synthetic Dataset

The main algorithm begins with user interaction, where the selection of a product of interest prompts the algorithm to retrieve relevant reviews for analysis. These reviews undergo a meticulous preprocessing phase, which includes tokenization, lemmatization, part-of-speech tagging, and the removal of stopwords, punctuation, and non-alphanumeric characters. This process ensures that the text is distilled to its most meaningful components, facilitating a more accurate semantic analysis.

The culmination of the preprocessing and semantic analysis stages is the calculation of a weighted average semantic similarity score for each tweet. This score provides a quantifiable measure of the tweet's relevance to the identified safety concerns. The algorithm further aggregates these scores to derive an overall safety assessment for the product in question.

To evaluate the performance of the classification models, the Precision-Recall (PR) curves and ROC curves are employed. The PR curves illustrate the trade-off between precision and recall, especially in datasets with a significant imbalance between classes. Interpolating these curves provides a smoother representation, highlighting the model's performance across different levels of recall. The ROC curves, on the other hand, plot the True Positive Rate (TPR) against the False Positive Rate (FPR), providing insight into the model's ability to discriminate between classes at various threshold settings. The Area Under the Curve (AUC) summarizes the overall performance of the classification system.

Before forwarding into the detailed analysis of the Precision-Recall and ROC curves, it is essential to examine the performance metrics at various threshold levels. The following Table 5 presents the best values achieved per metric (accuracy, precision, recall, and F1 score) for each semantic similarity measure:

Measure	Accuracy	Precision	Recall	F1 Score
Wu-Palmer (wup) [23]	0.75	0.71	1.00	0.83
Resnik (res) [24]	0.75	0.71	1.00	0.83
Jiang-Conrath (jcn) [25]	0.75	0.71	1.00	0.83
Lin [27]	0.75	0.71	1.00	0.83
Path [29]	0.75	0.71	1.00	0.83
Leacock-Chodorow (lch) [26]	**0.88**	**0.83**	**1.00**	**0.91**

Table 5: Performance metrics for each semantic similarity measure in Synthetic dataset.

Across different thresholds, the measures Wu-Palmer, Resnik, Jiang-Conrath, Lin, and Path similarities consistently achieved an accuracy of 0.75, precision of 0.71, recall of 1.0, and an F1 score of 0.83. These results indicate a high level of performance across these measures but also reveal a uniformity in their effectiveness, suggesting that none of these measures significantly outperformed the others. However, the Leacock-Chodorow (lch) measure stands out by achieving the highest performance metrics. It attained an accuracy of 0.88, precision of 0.83, recall of 1.0, and an F1 score of 0.91.

Below,it is presented the comparative interpolated Precision-Recall curves' diagram generated from the algorithm's performance on the synthetic dataset. This comparative analysis aims to underscore our model's capability in distinguishing between classes with varying degrees of sensitivity and specificity, thus reflecting on its utility and reliability in practical applications.

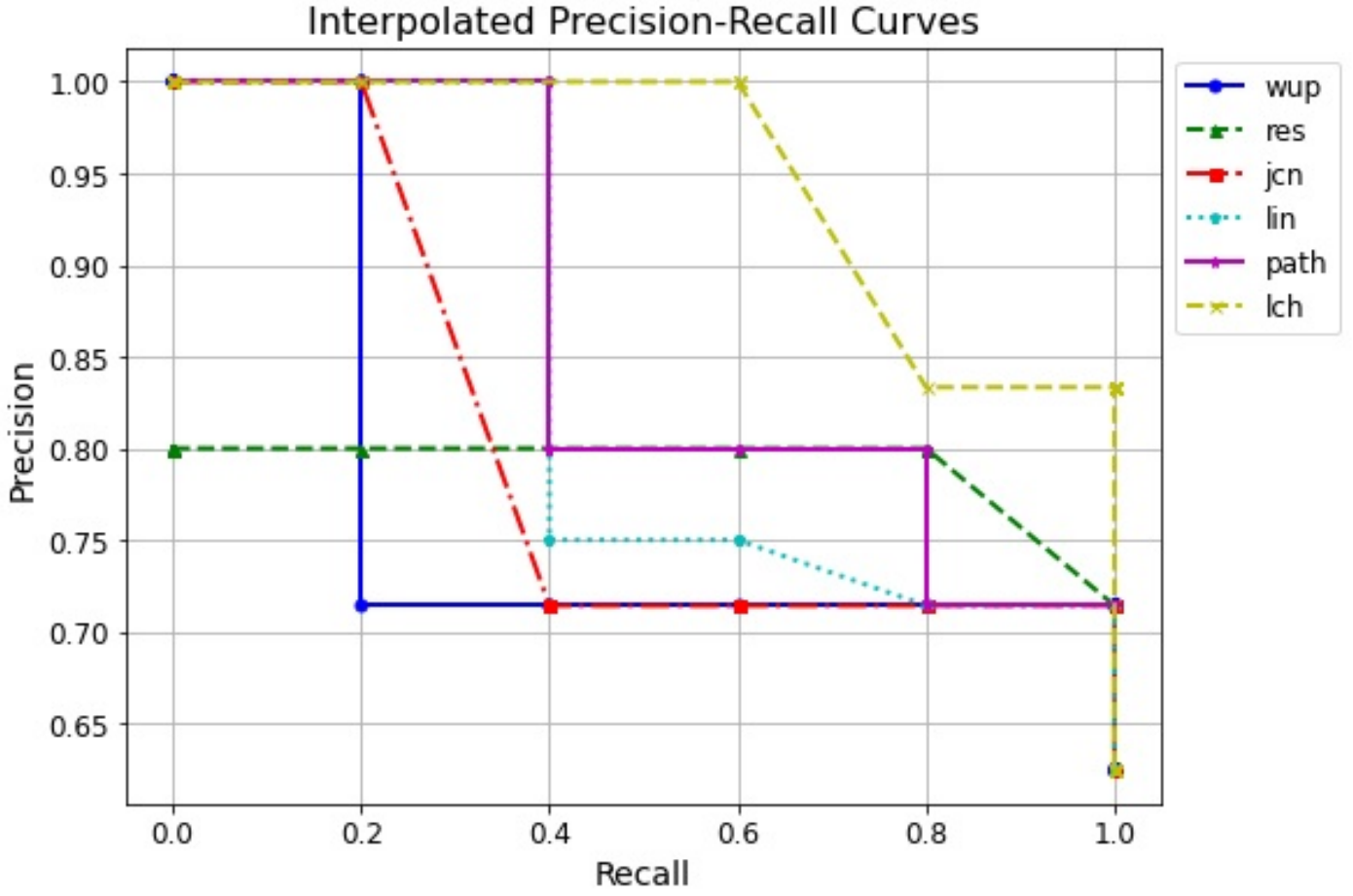


Figure 18: Interpolated Precision-Recall Curves.

Looking at the Interpolated Precision-Recall Curves18, it's evident that different semantic similiratity measures yield varied results. Each curve represents a distinct semantic similarity method applied to the task of sentiment analysis on tweets related to toy safety. Below are some points for analysis:

- **Wu & Palmer (WUP) Method:** The solid blue line remains at high precision until about 0.2 recall, indicating that this method maintains a high level of precision for a certain amount of retrieved instances but then sharply declines.
- **Resnik (RES) Method:** The dashed red line shows an immediate drop in precision as recall increases, suggesting that while it retrieves more instances, they are less likely to be relevant.
- **Jiang & Conrath (JCN) and Lin:** Both semantic similiratity measures (dotted green and dash-dot cyan lines) demonstrate a middle ground between precision and recall, but with fluctuations. They maintain moderate precision across different recall levels.
- **Path and Leacock-Chodorow (LCH):** These curves (solid purple and dashed yellow lines) exhibit a step-wise behavior, which may suggest a smaller set of high-confidence predictions. The precision remains constant over several intervals of recall, then drops to the next 'step'.

Precision-Oriented Approach: The Wu & Palmer (WUP) method, represented by the solid blue line, would be the preferred approach if precision is of utmost importance. This method shows a high level of precision up to a recall of about 0.2, which means it is very reliable in identifying dangerous toys when the certainty of the prediction is prioritized over the number of identified risks. A precision-oriented approach would be crucial if the implications of falsely identifying a toy as dangerous could lead to significant consequences, such as unwarranted recalls or legal actions. Therefore, the WUP method may be the best choice when the cost of false positives is high and the aim is to minimize the risk of incorrectly labeling a safe toy as dangerous.

Recall-Oriented Approach: Conversely, if the recall is more critical, suggesting that it is more important to identify as many potentially dangerous toys as possible, even at the risk of including safe toys in the danger category, then a different method may be more appropriate. For example, the Lin similarity approach, indicated by the black dash-dot line, shows a steadier decline in precision for an increase in recall, which implies a better balance. This approach might be preferred when the focus is on not missing any potential dangers, accepting that some non-dangerous toys might be flagged in the process.

Balanced Approach: For a balance between precision and recall, the semantic similarity measures represented by the dotted green (JCN) line offer moderate precision across different levels of recall. The JCN measure might be appropriate if the interest party desires a compromise between identifying most of the dangerous toys and maintaining a reasonable level of confidence in the results.

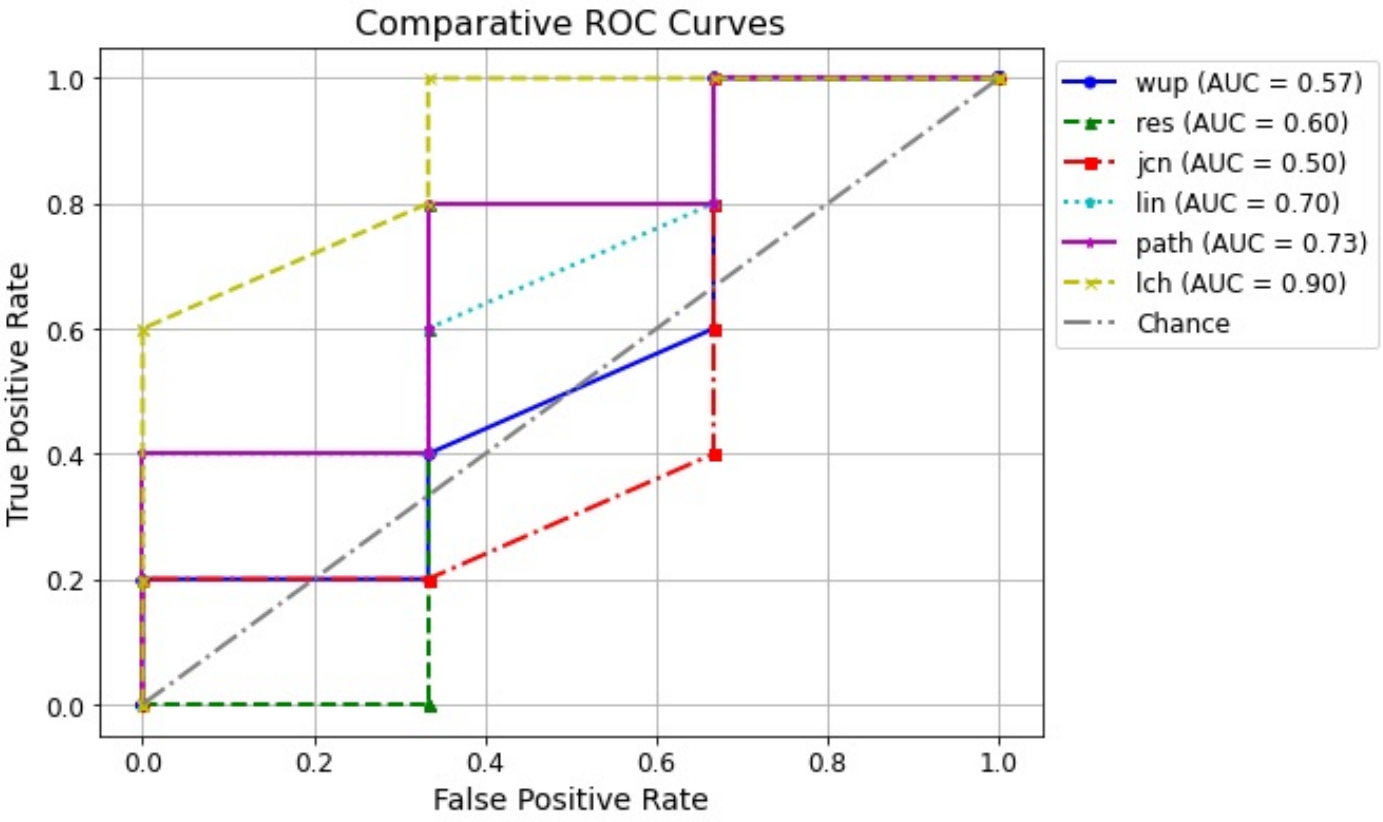


Figure 19: Comparative ROC Curves.

- **Wu & Palmer (WUP) Method:** The ROC curve for the WUP method, with an AUC of 0.57, suggests a performance close to random chance. This indicates a less effective method for the classification task at hand, as it only slightly improves the ability to differentiate between safe and dangerous toys over random guessing.
- **Resnik (RES) Method:** The RES method has an AUC of 0.60, which points to a modest ability to classify tweets correctly. Its performance is slightly better than the WUP method, but still not significantly higher than random chance, suggesting limited applicability for safety-critical applications.
- **Jiang & Conrath (JCN) Method:** With an AUC of 0.50, the JCN method's performance is equivalent to random guessing. This method does not provide a reliable classification of the tweets and is not recommended for this particular task.
- **Lin Method:** The Lin method shows a substantial improvement with an AUC of 0.70. This value indicates a better-than-random ability to discriminate between the positive and negative classes and shows promise for this application.
- **Path Method:** Achieving an AUC of 0.73, the Path method is shown to have a robust classification capability, suggesting that it can reliably identify dangerous toys based on tweet analysis.
- **Leacock-Chodorow (LCH) Method:** The LCH method exhibits superior performance with an AUC of 0.90, indicating a high level of effectiveness in classifying tweets accurately. This method stands out as the most proficient among the evaluated options.

In conclusion the LCH method is highly recommended for an interest party's application due to its significant ability to correctly classify tweets while minimizing the false positive rate. The Path and Lin measures also present viable options, with respectable AUC values indicating their effectiveness in distinguishing between classes.

4.4.2 Application on the Amazon Dataset

The next application of the main algorithm involves the Amazon dataset. Specifically, users are given the option to select from a menu of ASINs, which are unique identifiers used by Amazon to distinguish products. Therefore, by selecting an ASIN, the user can examine whether a product is dangerous or not, based on the analysis of user reviews. Similar to the synthetic dataset, first, the table which includes the results for all the metrics per measure is presented. The following diagramms include the comparative Precision-Recall curve for all measures is presented, followed by the comparative diagram for the ROC curves.

Measure	Accuracy	Precision	Recall	F1 Score
Wu-Palmer (wup) [23]	1.00	0.75	0.86	0.88
Resnik (res) [24]	0.75	0.75	0.75	0.75
Jiang-Conrath (jcn) [25]	1.00	0.75	0.86	0.88
Lin [27]	0.80	1.00	0.89	0.88
Path [29]	1.00	0.75	0.86	0.88
Leacock-Chodorow (lch) [26]	**0.80**	**1.00**	**0.89**	**0.88**

Table 6: Best metric values for each semantic similarity measure in the Amazon dataset.

From the above data and Table 6 , it is concluded that the Wu-Palmer (wup) measure, the Jiang-Conrath (jcn) measure and the Path measure a show high accuracy (1.00) and high F1 Score (0.88). Moreover, the Lin and Leacock-Chodorow (lch) measures have the highest precision and best recall with an F1 Score of 0.88, but their accuracy is lower at 0.80.

Based on the results and particularly considering the F1 Score, which is a combination of precision and recall, the Leacock-Chodorow (lch) and Lin measures seem to be the best, as they combine good precision (1.00), high recall (0.89), and very good F1 Score (0.88). Having to choose one, Leacock-Chodorow (lch) might be slightly preferred due to its better accuracy at specific thresholds.

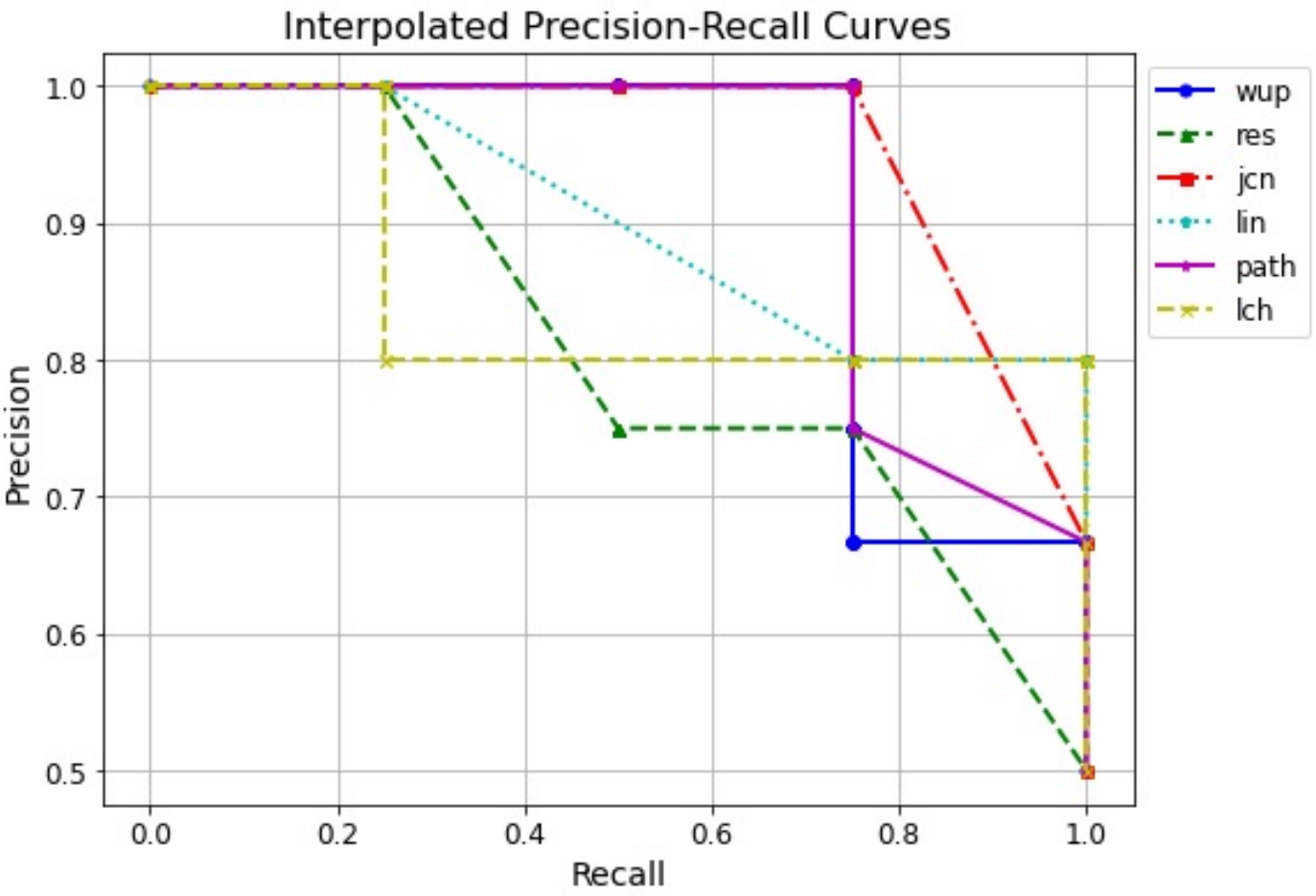


Figure 20: Comparative Precision-Recall Curve for the Amazon Dataset

The interpolated precision-recall curves 20 for the AMAZON dataset confirm the aforementioned results, revealing that the 'lin' and the 'lch' measures demonstrate better performance, maintaining high precision even at high recall levels. This indicates their ability to retrieve relevant items with minimal irrelevant results, making it a suitable choice for applications where precision is paramount.

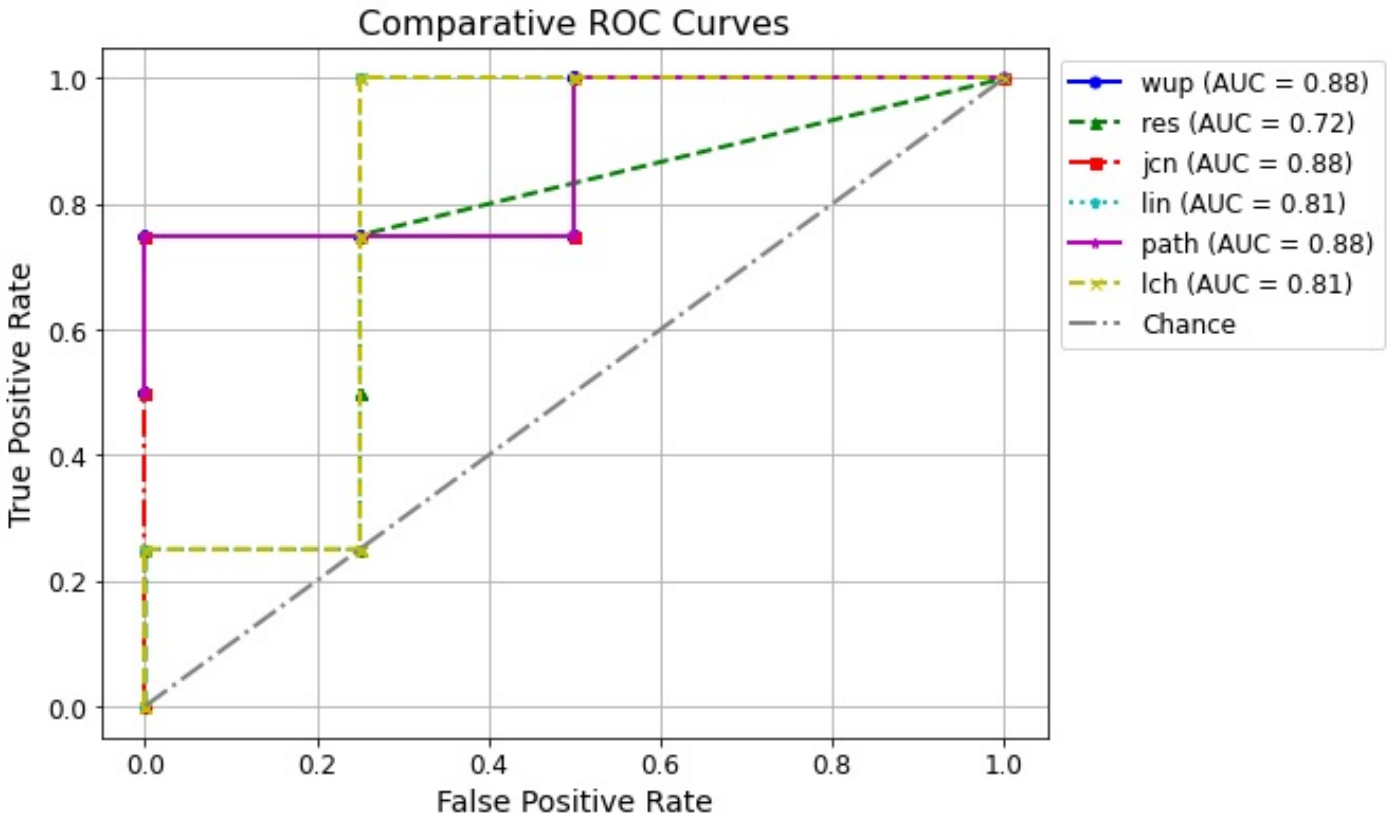


Figure 21: Comparative ROC Curve Diagram for the Amazon Dataset

The presented ROC curves 21 illustrates the performance of different semantic similarity methods (res, wup, jcn, lin, path, lch) in a binary classification task for "Dangerous" and "Safe" categories. In this particular case, all the measures except 'res', show verz good performance with'wup', 'jcn', and 'path' having AUC values of 0.88, and the 'lin' and 'lch' methods with AUC values of 0.81. The 'res' method performs the least well, with an AUC of 0.72, suggesting a more limited ability to differentiate between the classes compared to the other methods.

When comparing the results from the interpolated precision-recall curves across the two datasets, it becomes evident that the "lch" method consistently outperforms other semantic similarity measures. The same way, regarding the AMAZON dataset, the "lch" method maintains exceptionally high precision even at high recall levels, demonstrating its superior ability to effectively balance both metrics in this context. This suggests that the effectiveness of semantic similarity measures can vary significantly depending on the specific dataset and task, highlighting the importance of careful evaluation and selection based on the desired application requirements.

4.5 Experiment 2: Results of the Advanced Algorithm using various Semantic Similarity Measures

The next experiment concerns the application of the Advanced algorithm to datasets. Briefly, as mentioned in the previous section the additions of the Advanced algorithm compared to the main algorithm focus on

categorization of reviews, advanced preprocessing, vectorization of reviews, filtering of non-vocabulary words, creation of analysis terms and processing analysis terms with sentiment analysis.

4.5.1 Application on Synthetic Dataset

Initially the new experiment starts by applying the Advanced algorithm to the Synthetic dataset. The results of the application of the Advanced Algorithm on the Synthetic Dataset are as follows in Table 7:

Measure	Accuracy	Precision	Recall	F1 Score
Wu-Palmer (wup) [23]	0.75	0.71	1.00	0.83
Resnik (res) [24]	0.75	0.71	1.00	0.83
Jiang-Conrath (jcn) [25]	0.75	0.71	1.00	0.83
Lin [27]	0.75	0.71	1.00	0.83
Path [29]	0.75	0.71	1.00	0.83
Leacock-Chodorow (lch) [26]	**0.88**	**0.83**	**1.00**	**0.91**

Table 7: Performance metrics after the application of Advanced Algorithm for each semantic similarity measure in Synthetic dataset.

The tables present the performance metrics of various semantic similarity measures on the synthetic dataset, evaluating their accuracy, precision, recall, and F1-score at different threshold levels. The results indicate that the Leacock-Chodorow (lch) measure achieves the highest values for all metrics, outperforming the other similarity measures.

Specifically, the Leacock-Chodorow (lch) measure records the highest accuracy (0.88), precision (0.83), recall (1.0), and F1-score (0.91). These results demonstrate that this particular measure has the best performance among the tested measures, successfully balancing both precision and recall.

However, it is important to note that despite the application of the advanced algorithm, there were no significant changes in the performance metrics. This lack of variation can be attributed to the synthetic dataset's limited size and lack of diversity. The dataset does not present the variety needed to highlight the differences brought by the advanced algorithm. In contrast, in the next section, it will be observed that the advanced algorithm achieves different and improved results when applied to the more diverse and extensive Amazon dataset.

Moving to the presentation and commenting of the comparative Precision-Recall Curves, in Figure 22, it is obvious that the Leacock-Chodorow (lch) offers the highest and best balance between precision and recall.

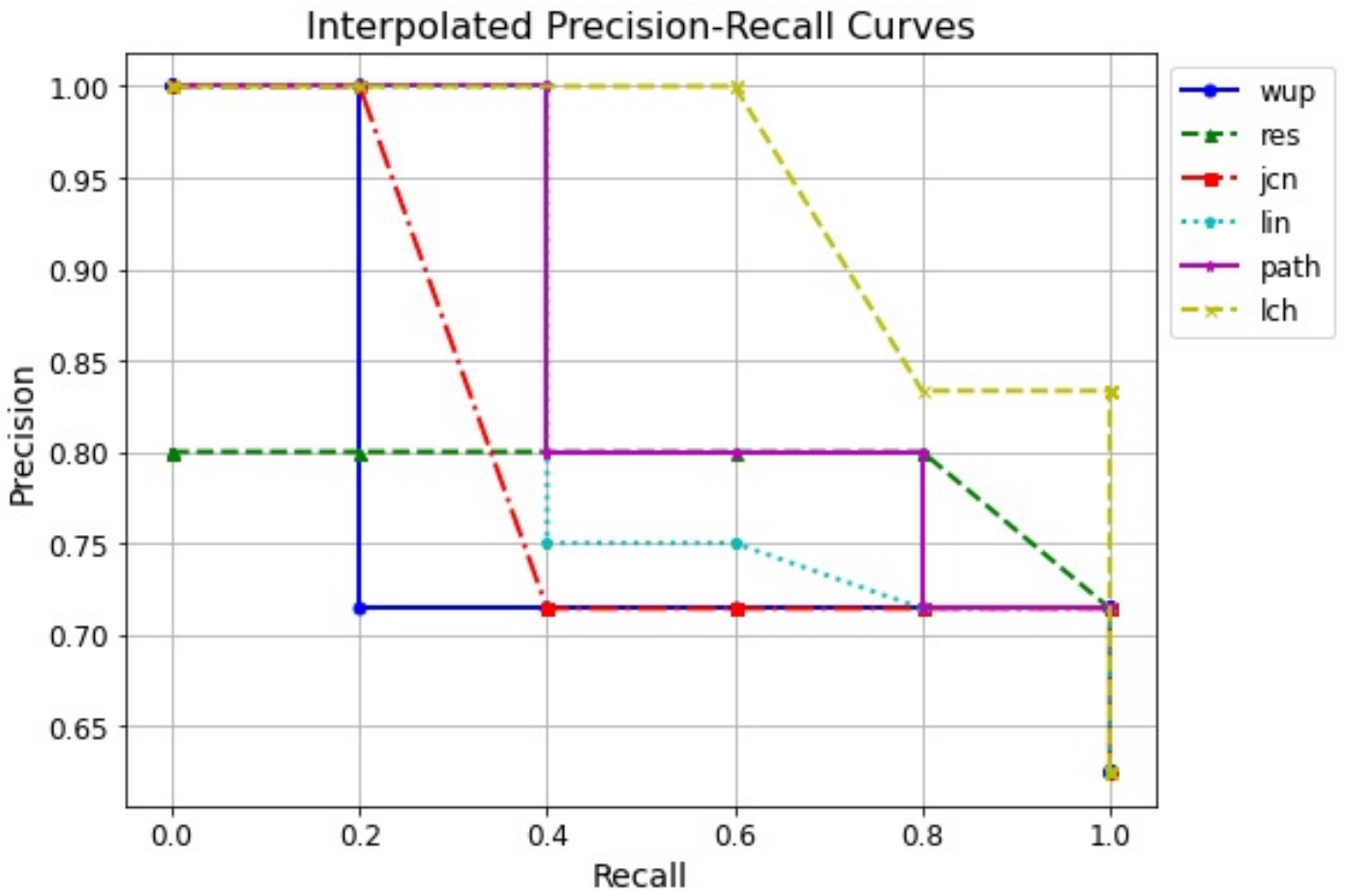


Figure 22: Comparative Precision-Recall Curves of the application of Advanced Algorithm on the Synthetic Dataset

In the next step, which concerns the analysion of ROC Curves in Figure 23 , it is also observed that the 'lch' method significantly outperforms the others, achieving the highest AUC of 0.90. This indicates that 'lch' has the best discriminatory power among the evaluated methods, effectively distinguishing between positive and negative instances. The 'path' and 'lin' methods also demonstrate good performance, with AUC values of 0.73 and 0.70 respectively. In contrast to the other Measures, the 'wup' and 'res' methods show moderate performance with AUCs of 0.57 and 0.60, suggesting a limited ability to differentiate between the classes. The 'jcn' method performs the poorest, with an AUC of 0.50, which is equivalent to random guessing (represented by the 'Chance' diagonal line).

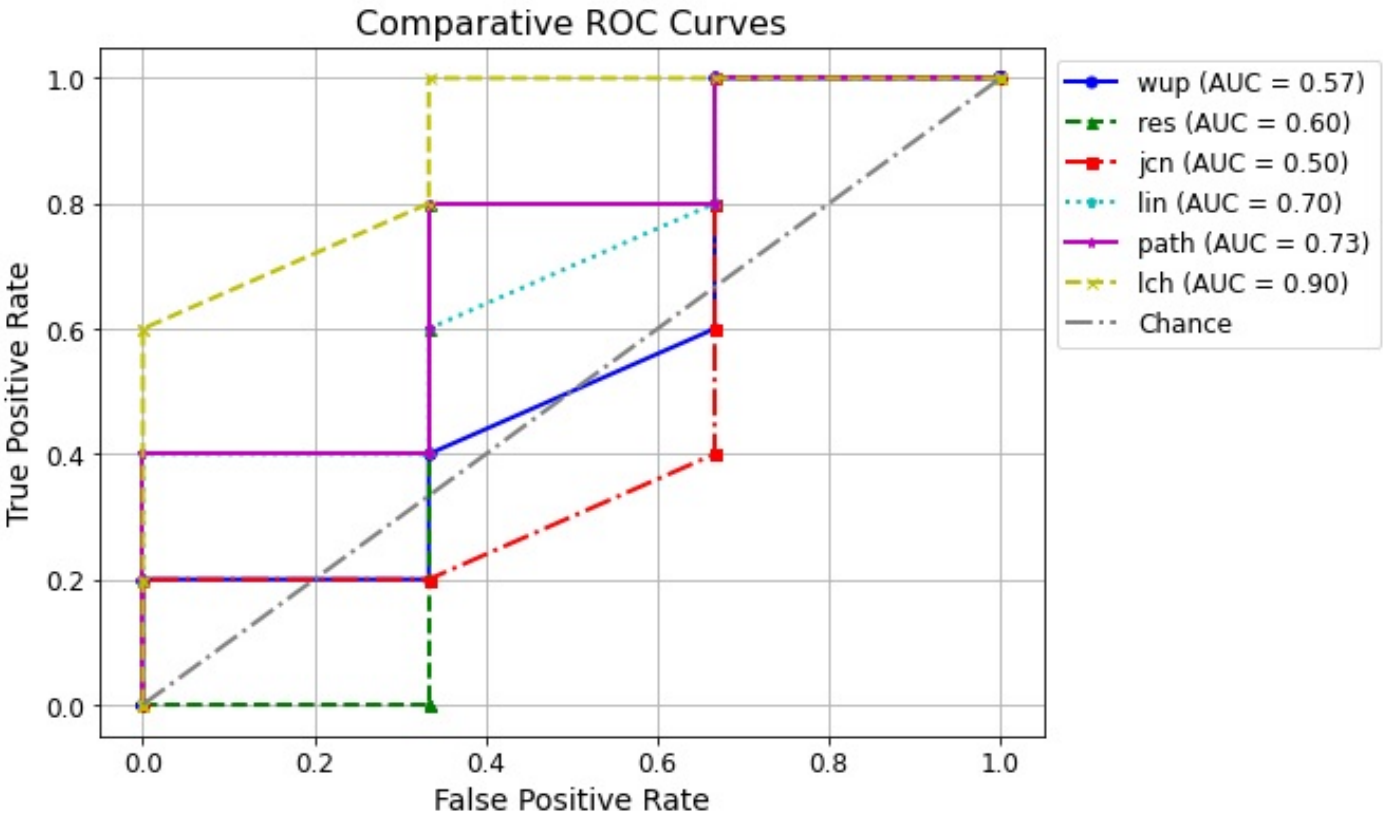


Figure 23: Comparative ROC Curves of the application of Advanced Algorithm on the Synthetic Dataset

4.5.2 Application on Amazon Dataset

Before proceeding to the presentation and the commenting of PR and ROC curves' diagrams, the results of the application of Advanced Algorithm for different thresholds are presented in the following Table **??**:

Measure	Accuracy	Precision	Recall	F1 Score
Wu-Palmer (wup) [23]	0.90	0.90	1.00	0.95
Resnik (res) [24]	0.90	0.90	1.00	0.95
Jiang-Conrath (jcn) [25]	0.90	0.90	1.00	0.95
Lin [27]	0.90	0.90	1.00	0.95
Path [29]	0.90	0.90	1.00	0.95
Leacock-Chodorow (lch) [26]	0.90	0.90	1.00	0.95

Table 8: Best metric values for each semantic similarity measure in the Amazon dataset using the Advanced Algorithm.

Upon reviewing the results, it is evident that the performance metrics across different measures are highly similar, with many measures achieving the same best values for Accuracy, Precision, Recall, and F1 Score. For instance, all measures reach a best Accuracy of 0.90, a best Precision of 1.00, a best Recall of 1.00, and a best F1 Score of 0.95. This uniformity indicates that the advanced algorithm is robust and effective across different measures when applied to the Amazon dataset.

Despite the advanced algorithm's application, the results exhibit minimal variation among the different measures. This outcome suggests that while the advanced algorithm is highly effective, the Amazon dataset's specific characteristics may not significantly differentiate the performance of various semantic similarity measures. The dataset's inherent structure and the nature of the reviews could be contributing factors to this observed uniformity.

However, among the measures, the Leacock-Chodorow (lch) measure stands out as the best performer based on its consistency in achieving the highest values across different measures, as resulted in Table 8 and in the previous applications of the other algorithms. This suggests that while all measures perform well, Leacock-Chodorow (lch) might be slightly more reliable in this context.

Below in Figure 24 , we observe that the performance of most measures is quite similar, suggesting that all measures perform comparably on the Amazon dataset. The Leacock-Chodorow (lch) measure, though, shows slightly better performance, remaining more stable in the high Recall region, indicating its effectiveness. The Wu-Palmer (wup) and Jiang-Conrath (jcn) measures also exhibit high Precision and Recall values, similar to Leacock-Chodorow. All measures demonstrate high stability in the high Recall region, suggesting that their performance does not significantly drop as Recall increases.

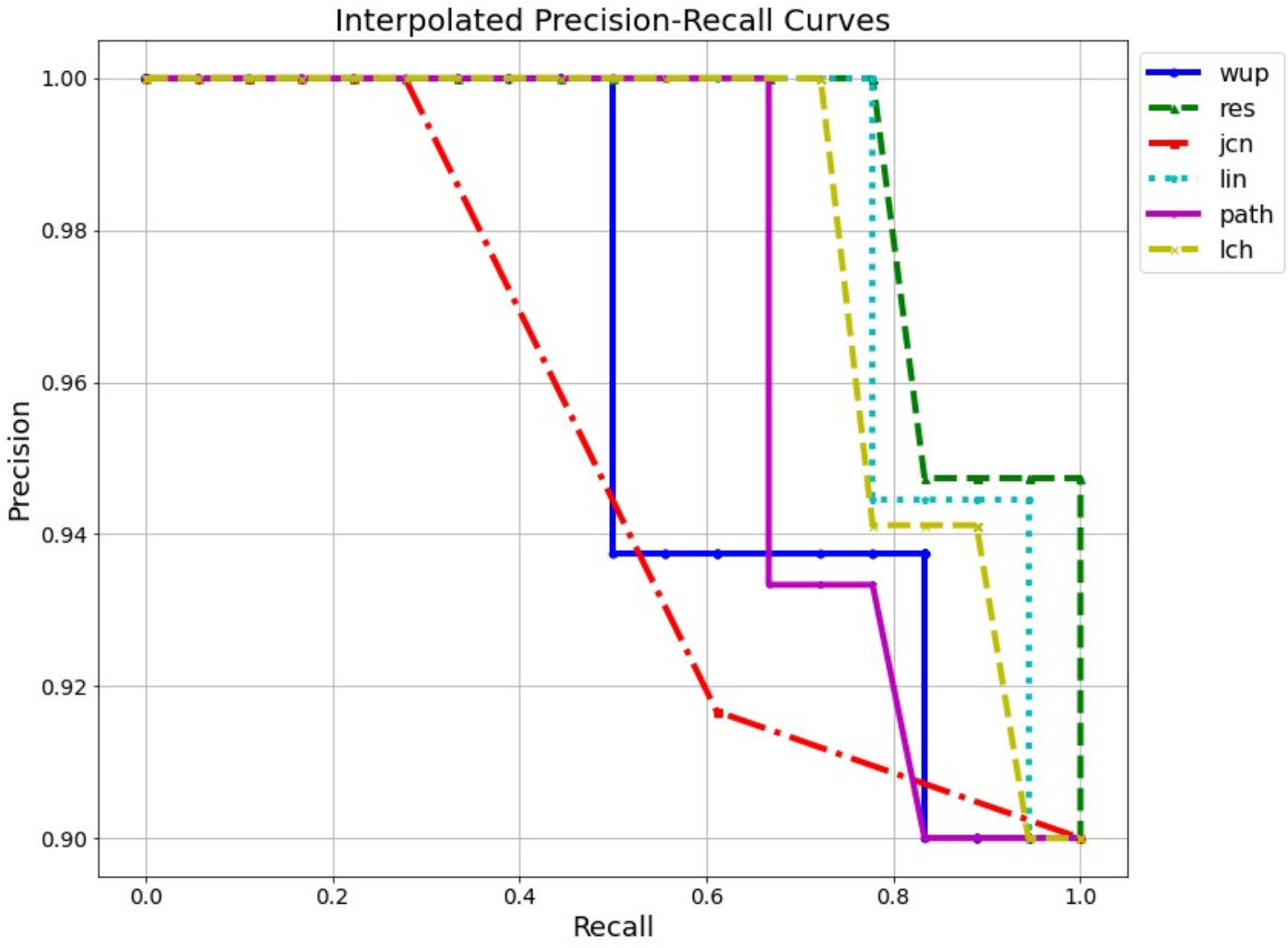


Figure 24: Comparative Precision-Recall Curves of the application of Advanced Algorithm on the Amazon Dataset

Comparing the interpolated precision-recall curves across the two datasets, we observe a notable similarity in the performance of the 'lch' method. This similarity indicate that the 'lch' method has the better performance, maintaining perfect precision up to a recall of approximately 0.8, indicating its superior ability to retrieve relevant items without sacrificing precision.

Similarly, as we see at the Comparative ROC Curves diagram in the Figure 25 below, the Resnik (res) measure achieves the highest Area Under the Curve (AUC) of 0.86, qualifying to the best performance in distinguishing between classes. This is followed very closely by Lin (lin) and Leacock-Chodorow (lch) measures, both with an AUC of 0.85, demonstrating their robustness in classification tasks.

The Path (path) measure shows a moderate performance with an AUC of 0.69, suggesting it is less effective compared to res, lin, and lch. The Wu-Palmer (wup) measure has an AUC of 0.67, indicating it performs reasonably well but is not as strong as the top measures. Finally, the Jiang-Conrath (jcn), with the lowest AUC of 0.49, performs slightly better than random guessing (AUC of 0.5), suggesting it is the least effective among the evaluated measures.

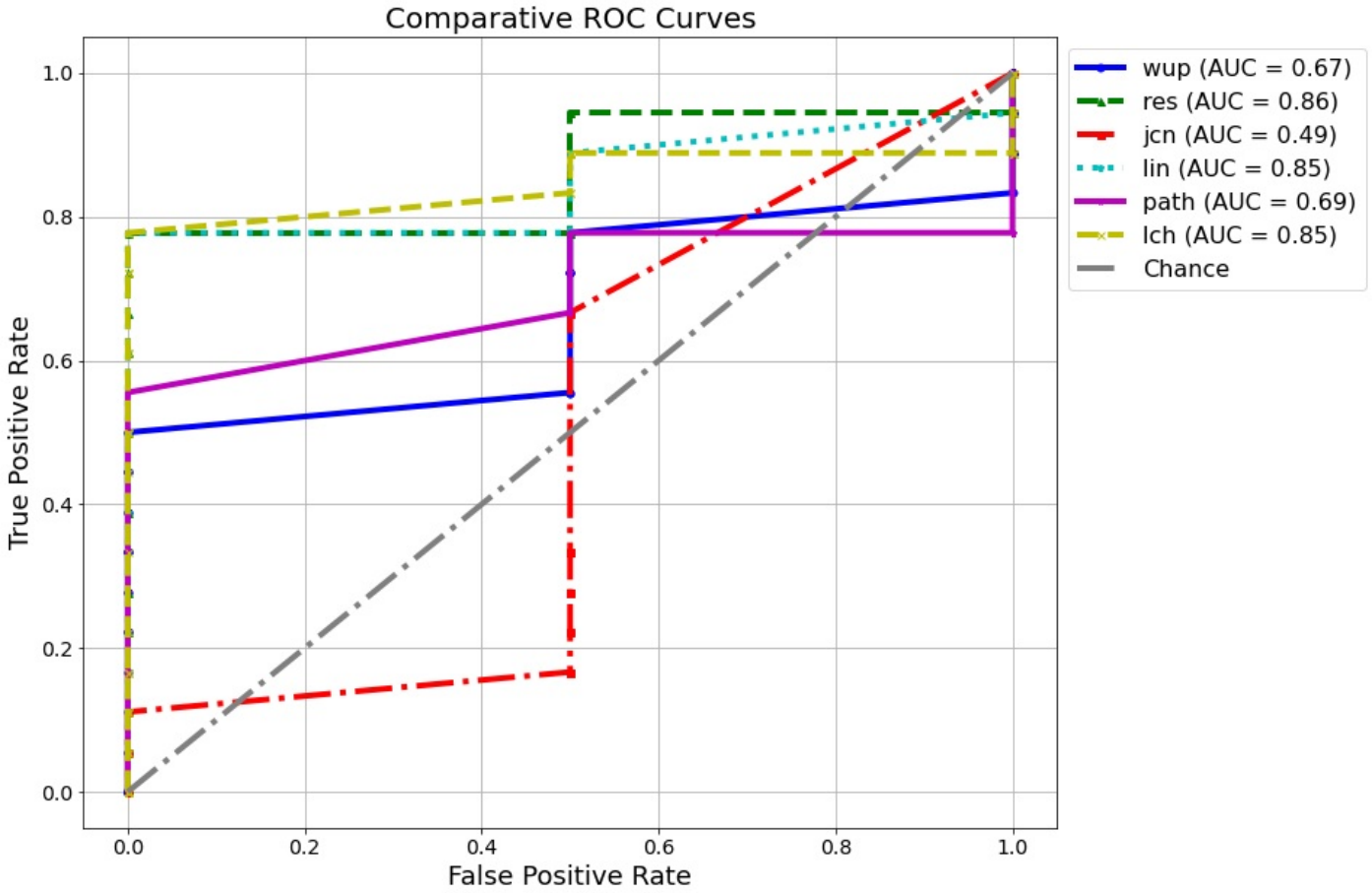


Figure 25: Comparative ROC Curves Diagram after the application of Advanced Algorithm on the the Amazon Dataset

4.6 Experiment 3: Results of the Ensemble Algorithm (Boosting) using various Semantic Similarity Measures

This section presents the results of applying the Ensemble Algorithm of different Semantics Similarity Measures (Boosting), which introduces a different approach to automating the creation of classes and assigning weights to semantic similarity (SemSim) scores. This methodology aims to address the challenge of ensuring consistent and adaptable weighting of similarity scores by dynamically adjusting weights based on the observed distribution of scores.

It is worth noting that the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) was primarily applied to the Amazon dataset due to the complexity of its data compared to the synthetic dataset. Specifically, the mathematical functions and operations present in the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) are more appropriate for more complex data that requires more processing, hence the need for class allocation and creation of corresponding weights.

4.6.1 Application on Amazon Dataset

The user initiates the execution of the program, and a selection menu of Amazon products appears for the user to choose the corresponding ASIN. An example of the menu display is shown in the following Table 9:

Number	ASIN
1	0020232233
2	038536539X
3	0486277577
4	0486402029
5	0486427706
6	0486448789
7	0545325234
8	0545346193
9	0545561647
10	0615638996
11	0641869665
12	0692770445
13	0735332258
14	0735331146
15	0735333467
16	0735351228
17	0735335109
18	0152014764
19	0769658237
20	0769663192
21	0769660835
22	0786950072

Table 9: Selection Menu for Amazon ASINs

In the first part of the algorithm, the maximum semantic similarity scores are calculated, resulting in a table of the following form 10:

ASIN	Review ID	Lemma	Keyword	Method	SemSim Score
0545325234	147	by	absolute	wup	0.5
0545325234	147	by	absolute	res	-1.0
0545325234	147	by	absolute	jcn	-1.0
0545325234	147	by	absolute	lin	-1.0
0545325234	147	by	absolute	path	0.3333
0545325234	147	by	absolute	lch	-1.0
0545325234	147	by	abuse	wup	0.4
0545325234	147	by	abuse	res	-1.0
0545325234	147	by	abuse	jcn	-1.0
0545325234	147	by	abuse	lin	-1.0
0545325234	147	by	abuse	path	0.25
0545325234	147	by	abuse	lch	-1.0

Table 10: Maximum Semantic Similarity Scores

In the next step, these semantic similarity scores are loaded to create classes and their corresponding weights. The Table 11 below shows the results from optimizing the classes along with the corresponding weights assigned to each class:

Method	Class Interval	Weight	Threshold
wup	(-1.0009, 0.222)	0.3327	0.5
wup	(0.2221, 0.4)	0.3099	0.5
wup	(0.4001, 0.5)	0.2838	0.5
wup	(0.5001, 1.0)	0.0736	0.5
res	(-0.0009, 4.04e-300)	0.7328	4.04e-300
res	(0.0001, 1.0)	0.2672	4.04e-300
jcn	(-0.0009, 0.0464)	0.3584	0.0678
jcn	(0.0465, 0.0678)	0.5347	0.0678
jcn	(0.0679, 1.0)	0.1069	0.0678
lin	(-0.0009, 0.0528)	0.2380	0.3311
lin	(0.0529, 0.0758)	0.2381	0.3311
lin	(0.0759, 0.224)	0.2381	0.3311
lin	(0.2241, 0.331)	0.2381	0.3311
lin	(0.3311, 1.0)	0.0476	0.3311
path	(-1.0009, 0.111)	0.2371	0.3333
path	(0.1111, 0.2)	0.2386	0.3333
path	(0.2001, 0.25)	0.2548	0.3333
path	(0.2501, 0.333)	0.2094	0.3333
path	(0.3331, 1.0)	0.0600	0.3333
lch	(-0.0009, 0.464)	0.6169	0.4642
lch	(0.4641, 0.571)	0.1908	0.4642
lch	(0.5711, 1.0)	0.1923	0.4642

Table 11: Optimized Classes and Weights for Semantic Similarity Measures

Finally, the classes and weights for each semantic similarity measure are assigned, and the algorithm is executed to produce the following results:

Measure	Accuracy	Precision	Recall	F1 Score
Wu-Palmer (wup) [23]	0.90	0.90	1.00	0.95
Resnik (res) [24]	0.90	0.90	1.00	0.95
Jiang-Conrath (jcn) [25]	0.90	0.90	1.00	0.95
Lin [27]	0.92	0.92	1.00	0.96
Path [29]	0.90	0.90	1.00	0.95
Leacock-Chodorow (lch) [26]	**0.95**	**0.95**	**1.00**	**0.97**

Table 12: Best metric values for each semantic similarity measure in the Amazon dataset using the Ensemble Algorithm of different Semantics Similarity Measures (Boosting).

The Table 12 above present the performance metrics for various semantic similarity measures applied to the Amazon dataset using the Ensemble Algorithm of different Semantics Similarity Measures (Boosting). Similar to the previous experiments, the Leacock-Chodorow (lch) achieves the highest performance, with an F1 Score of 0.97, demonstrating again superior performance in this context.

The interpolated precision-recall curves in the diagram 26 show that the Resnik (res), Lin (lin), and Leacock-Chodorow (lch) consistently achieve high Precision and Recall across various thresholds, indicating strong performance. In contrast, Wu-Palmer (wup) shows a significant drop in Precision as Recall increases, suggesting a trade-off where high Recall leads to lower Precision. Path (path) exhibits similar behavior to Wu-Palmer, with Precision decreasing as Recall increases. Jiang-Conrath (jcn) shows the most variation, with significant drops in Precision at certain points, reflecting inconsistent performance.

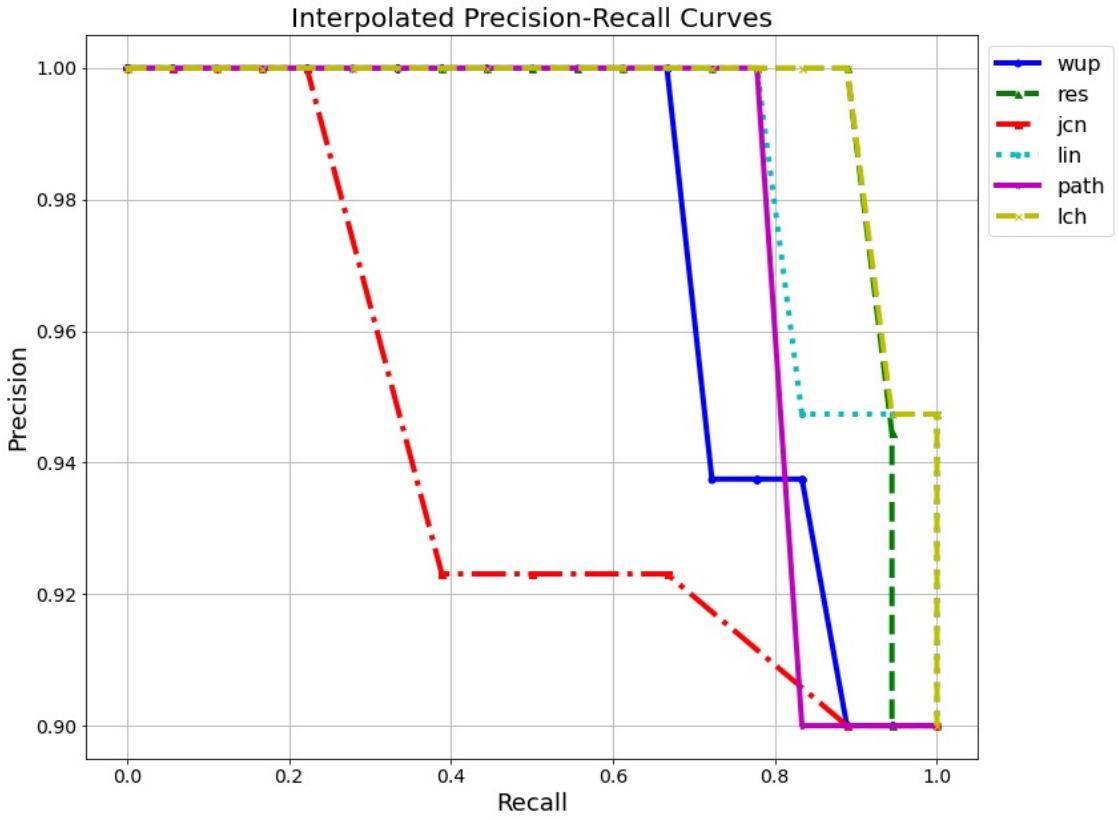


Figure 26: Interpolated Precision-Recall Curves after the application of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) on the Amazon Dataset

Examining the Comparative ROC Curves' diagram 27 created from the application of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting), as in the previous experiment, the Leacock-Chodorow achieves the highest AUC of 0.96, indicating excellent performance in distinguishing between the classes. Resnik (res) Lin (lin) follows with an AUC of 0.92, also showing strong performance. Lin (lin) has an AUC of 0.90, which is slightly lower than Resnik but still indicates top performance. Wu-Palmer (wup) has a moderate AUC of 0.78, showing decent performance but not as strong as Resnik, Lin, or Leacock-Chodorow. Path (path) has an AUC of 0.81, indicating reasonable performance. Finally, Jiang-Conrath (jcn) has the lowest AUC of 0.50, indicating poor performance.

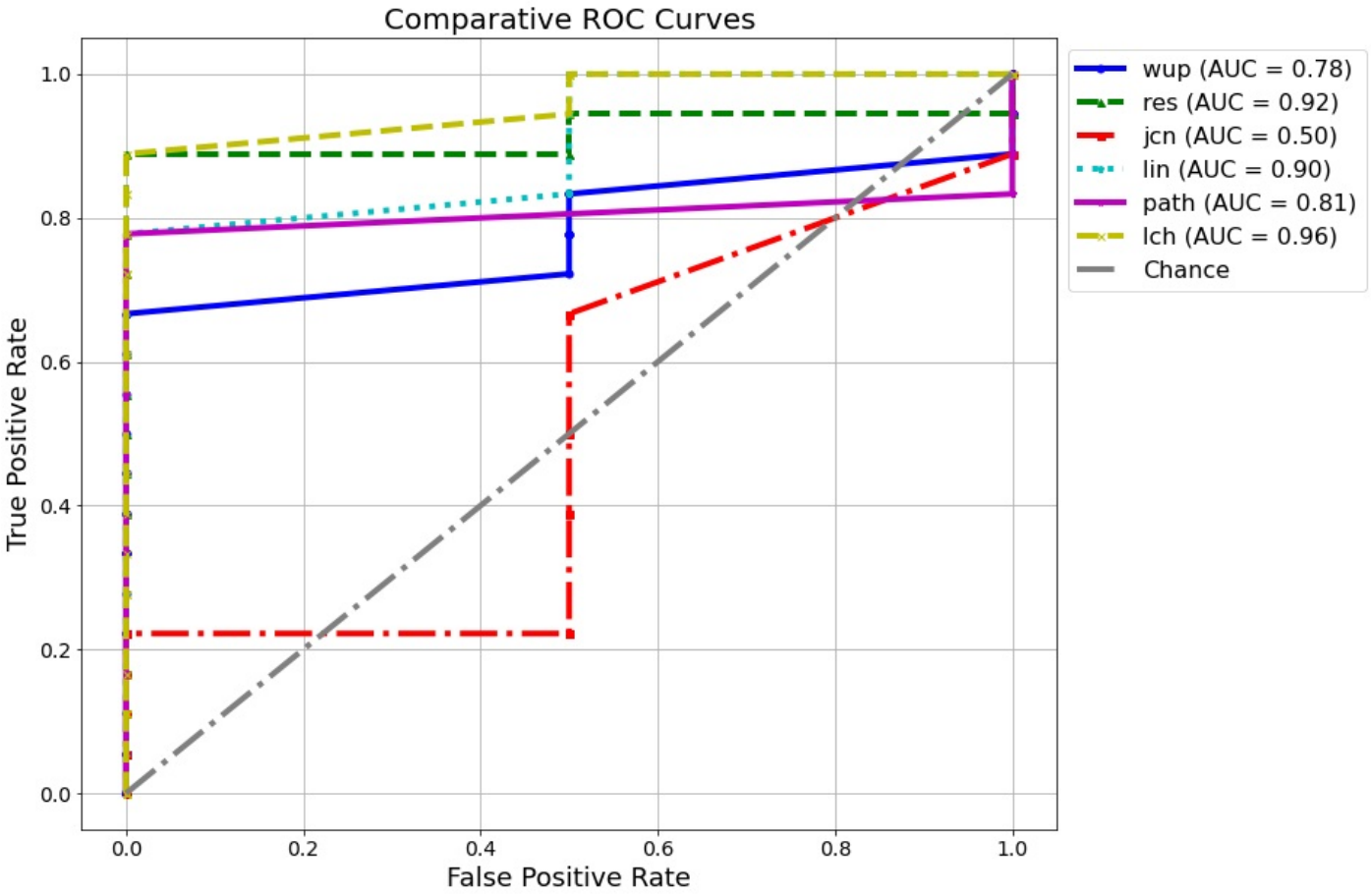


Figure 27: Comparative ROC Curves Diagram after the application of Ensemble Algorithm of different Semantics Similarity Measures (Boosting) on the Amazon Dataset

In summary, the application of the Ensemble Algorithm of different Semantics Similarity Measures (Boosting) showed that the Leacock-Chodorow measure shows the best overall performance, with high AUC values and consistent Precision-Recall metrics, indicating their effectiveness in the classification task. The Resnik and Lin measures also performs very well, maintaining its top performance, but with slightly lower than Leacock-Chodorow. In contrast, Wu-Palmer and Path measures show moderate performance, while Jiang-Conrath performs poorly compared to the others.

4.7 Performance Evaluation of BERT and VADER Models

The section below details the performance outcomes of the BERT and VADER models when applied to the Amazon dataset. These models were evaluated alongside the semantic similarity measures to provide a comprehensive comparison. The performance metrics considered include accuracy, precision, recall, and F1 score, and are presented in Table 13.

Algorithm	Accuracy	Precision	Recall	F1 Score
Simple Algorithm (Leacock-Chodorow (lch))	0.80	1.00	0.89	0.88
Advanced Algorithm (Leacock-Chodorow (lch))	0.90	0.90	1.00	0.95
Boosting Algorithm (Leacock-Chodorow (lch))	**0.95**	**0.95**	**1.00**	**0.97**
BERT	0.89	0.97	0.89	0.93
VADER	0.84	0.86	0.96	0.91

Table 13: Performance Metrics for all the algorithms, including BERT and VADER, on the Amazon dataset.

The comparative results indicate that the Boosting Algorithm (using Leacock-Chodorow measure) outperforms other algorithms, achieving the highest F1 Score of 0.97. BERT follows with an F1 Score of 0.93, showcasing its strong performance in sentiment analysis tasks. VADER, while not as precise as BERT, still delivers a commendable F1 Score of 0.91, validating its efficiency in sentiment analysis tasks.

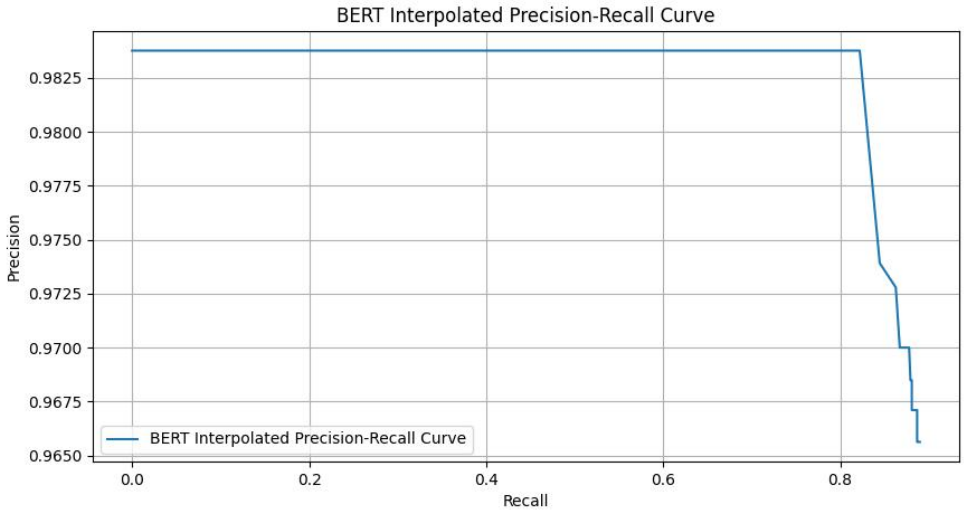


Figure 28: BERT Interpolated Precision-Recall Curve

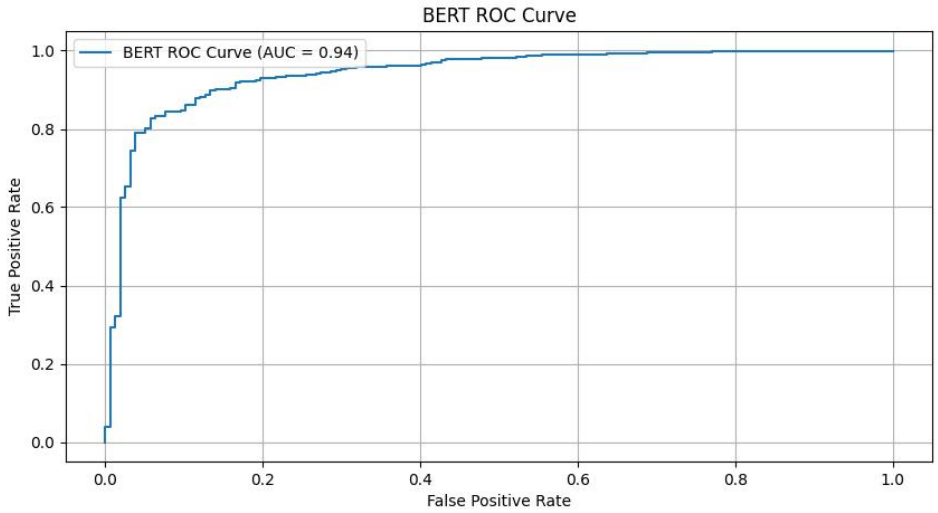


Figure 29: BERT ROC Curve

BERT Analysis: The BERT model's performance, as shown in Figure 28, demonstrates an impressive ability to maintain high precision across a wide range of recall values. This indicates that BERT is highly effective at correctly identifying relevant instances without sacrificing too much precision, even as the recall increases. The ROC curve in Figure 29 further supports this, with an AUC of 0.94, indicating excellent discriminatory power between positive and negative classes.

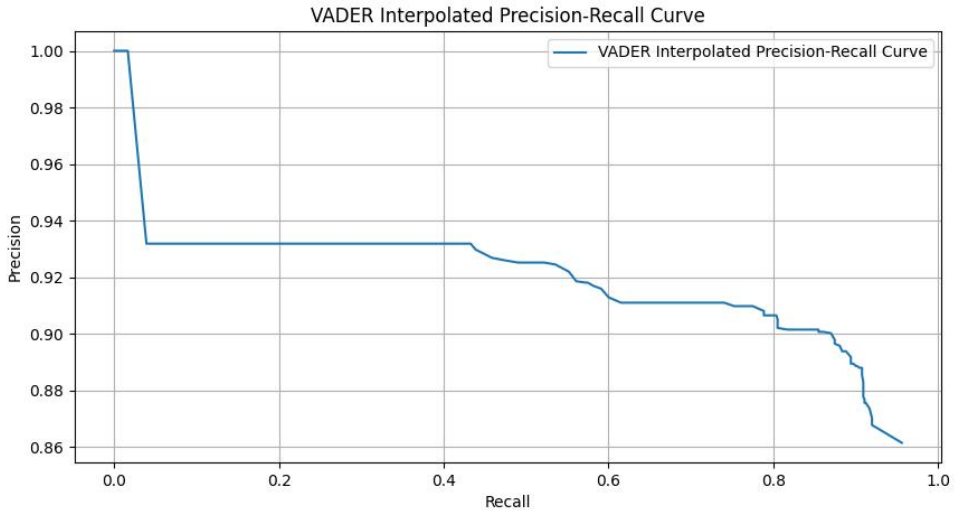


Figure 30: VADER Interpolated Precision-Recall Curve

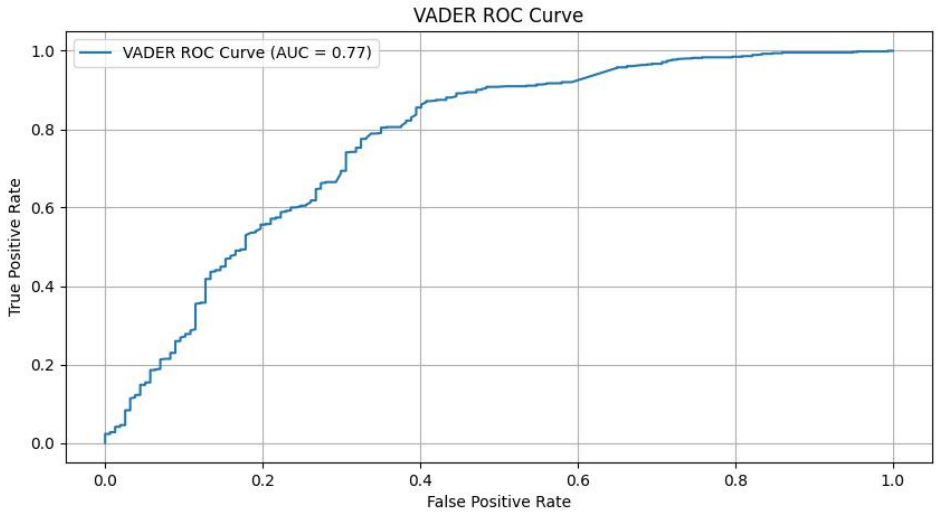


Figure 31: VADER ROC Curve

VADER Analysis: The VADER model, depicted in Figure 30, shows strong performance with high precision at lower recall levels. However, there is a noticeable decline in precision as recall increases, which indicates some trade-offs in identifying relevant instances. The ROC curve in Figure 31 shows an AUC of 0.77, reflecting good but not exceptional discriminatory power. VADER's strength lies in its efficiency and speed, providing a practical option for scenarios where rapid sentiment analysis is required, despite its lower performance compared to BERT.

While traditional semantic similarity measures can be highly effective, advanced NLP models such as BERT

offer significant advantages in handling nuanced and complex sentiment analysis scenarios. VADER, despite being a simpler model, proves to be a valuable tool for rapid sentiment analysis, particularly when computational efficiency is prioritized. The findings support the integration of these models with traditional semantic similarity measures to achieve comprehensive and accurate product categorization.

5 Discussion and Future work

In the preceding chapters, various semantic similarity measures and their applications to synthetic and real-world datasets were explored. The initial application of these measures on a synthetic dataset allowed for controlled and insightful understanding of their performance. The findings from this experiment underlined the potential of semantic similarity measures in categorizing product safety based on user-generated reviews. However, the relatively uniform nature of the synthetic dataset limited the diversity of the results.

Following this, these measures were applied to the Amazon dataset, providing a more complex and varied set of user reviews. These experiments highlighted the strengths and weaknesses of different semantic similarity measures in a real-world context. The results indicated that the Leacock-Chodorow (lch) measure consistently outperformed other measures across various metrics, particularly in terms of F1 Score, showcasing its robustness in handling diverse and nuanced data from Amazon reviews, as shown in the following Figure.

Figure 35 illustrates the Comparative ROC Curves for five different algorithms applied to the Amazon dataset: the Simple Algorithm, the Advanced Algorithm, the Boosting Algorithm, BERT, and VADER. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

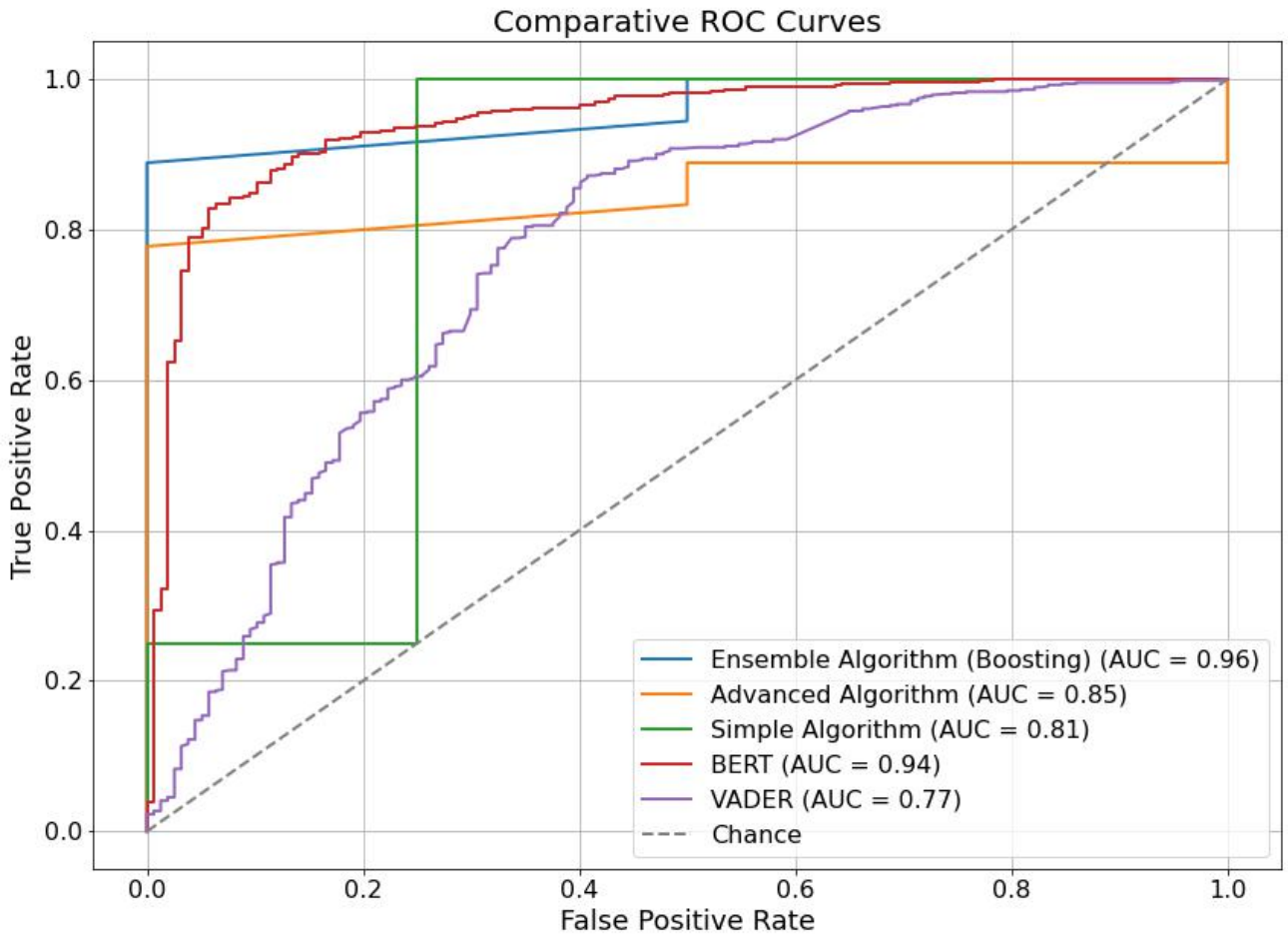


Figure 32: Comparative ROC Curves for Simple Algorithm, Advanced Algorithm, Ensemble Algorithm (Boosting), BERT, and VADER. The area under the curve (AUC) values are 0.81, 0.85, 0.96, 0.94, and 0.77 respectively, indicating the performance of each algorithm. The dashed line represents the performance of a random classifier (AUC = 0.50).

The performance of BERT and VADER was evaluated alongside the semantic similarity measures on the Amazon dataset. BERT, with its deep learning capabilities, demonstrated high accuracy and robustness in sentiment analysis, capturing the nuances in user reviews. VADER, while simpler, provided quicker and effective sentiment analysis. The inclusion of these models in the comparative analysis, like the Comparative ROC diagram, helped highlight the strengths and potential applications of advanced NLP techniques in market surveillance and product safety categorization.

More specifically, the ROC curves show that the Boosting Algorithm (with Leacock-Chodorow been qualified among the other SemSim measures) outperforms the other algorithms, achieving an AUC (Area Under the Curve) of 0.96. This indicates a high level of accuracy in distinguishing between the positive and negative classes. The BERT follows with an AUC of 0.94, the Advanced Algorithm has an AUC of 0.85, the Simple Algorithm has an AUC of 0.81 and VADER has an AUC of 0.77.

Additionally, an advanced algorithm, GPT-3.5 Turbo, was implemented to assess its capability in market surveillance tasks, as presented in the following chapters. The application of GPT-3.5 Turbo on the Amazon dataset yielded significantly better results than traditional semantic similarity measures. The model maintained high accuracy, precision, recall, and F1 score across different thresholds, demonstrating its advanced natural language processing capabilities and its potential for detailed textual analysis.

5.1 Future Work

Looking ahead, there are several avenues for further research and development based on the current findings:

5.2 Algorithm for Classifying Products Based on User Reviews Using Large Language Models (LLMs)

The following algorithm utilizes large language models (LLMs) to classify products based on user reviews or to generate outputs based on user queries. The input to the algorithm is a set of user reviews, and the output is determined by the specific query posed by the user.

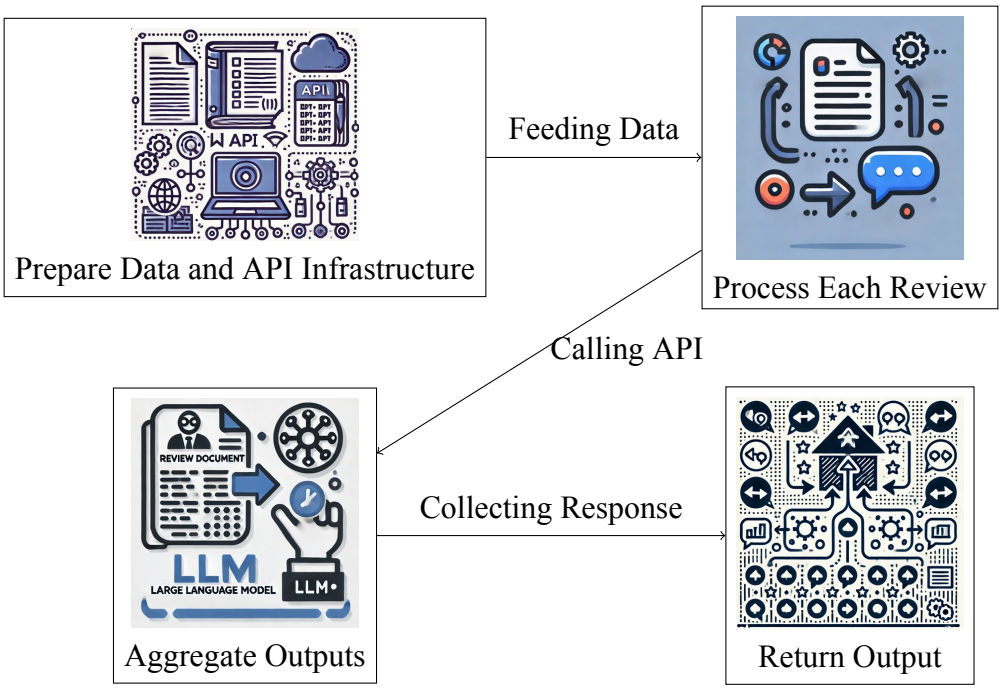


Figure 33: Visual Representation of the Algorithm for Classifying Products Based on User Reviews Using LLMs

5.2.1 Detailed Explanation of the Algorithm

A comprehensive explanation of the algorithm for classifying products based on user reviews using large language models (LLMs) is provided in the following section. This explanation includes the preparation of data and API infrastructure, processing of each review, aggregation of outputs, and returning the final output.

Step 1: Preparing Data and API Infrastructure

The necessary infrastructure for accessing the LLM API is set up, including building headers and data structures. The headers H define the content type and any required authorization, while the data D includes the structured user query Q and the list of reviews R.

$$\begin{aligned} H &= \text{Build Headers}() \\ D &= \text{Build Data}(Q, R) \end{aligned} \tag{37}$$

Step 2: Processing Each Review

For each review r in the list of reviews R, a prompt P is created by combining the user query Q with the review r.

$$P = \text{Create Prompt}(Q, r) \tag{38}$$

The prompt P is then sent to the LLM API along with headers H and data D, and the response A is received.

$$A = \text{Send Prompt and Receive Response}(P, H, D) \tag{39}$$

The output O is processed from the response A.

$$O = \text{Process Output}(A) \tag{40}$$

Step 3: Aggregating Outputs

The individual outputs O from each review are aggregated to form the final result based on the user query.

$$O_{\text{final}} = \text{Aggregate Outputs}(O) \tag{41}$$

Step 4: Returning Output

The aggregated output O_{final} is returned as the final result.

$$\text{Return } O_{\text{final}} \tag{42}$$

5.2.2 Symbols and Definitions

The following list defines the symbols used in the algorithm for classifying products based on user reviews using large language models (LLMs):

- H: Headers used for accessing the LLM API, defining the content type and any required authorization.
- D: Data structure that includes the user query Q and the list of reviews R.
- Q: User query that defines the desired output from the LLM.
- R: List of user reviews to be analyzed by the LLM.
- P: Prompt created by combining the user query Q with an individual review r.
- A: Response received from the LLM after sending the prompt P.
- O: Output processed from the response A.
- O_{final}: Aggregated output from all individual outputs O based on the user query.

5.2.3 Pseudocode for the Algorithm 4

Algorithm 4 Classifying Products Based on User Reviews Using Large Language Models

Input:
  R: List of user reviews.
  Q: User query defining the desired output.
Output:
  O: Output based on user query.
Procedure:
1. Preparing Data and API Infrastructure:
  Set up the necessary infrastructure for accessing the LLM API, including building headers and data structures.
  Headers: Define content type and authorization if needed.
  Data: Structure the user query and reviews for LLM processing.
  $H \leftarrow$ Build Headers()
  $D \leftarrow$ Build Data(Q, R)
2. Processing Each Review:
for each review r in R **do**
    $P \leftarrow$ Create Prompt(Q, r)
    $A \leftarrow$ Send Prompt and Receive Response(P, H, D)
    $O \leftarrow$ Process Output(A)
end for
3. Aggregating Outputs:
  Combine individual outputs O into a final result based on the user query.
4. Returning Output:
  Return the aggregated output O.

5.2.4 Experiment 4: Results of the Application of GPT-3.5 Turbo on Amazon Dataset

The final experiment involves the application of the developed algorithm to incorporate Large Language Models (LLMs), specifically OpenAI's GPT-3.5 Turbo, in market surveillance. This experiment was conducted on the Amazon dataset, aiming to investigate how GPT-3.5 Turbo categorizes products identified by ASIN based on user reviews. Initially, the results for different thresholds are summarized in the table below:

Threshold	Accuracy	Precision	Recall	F1 Score
0.0	0.7368	0.7368	1.0	0.8485
0.25	0.9474	0.9333	1.0	0.9655
0.5	0.9474	0.9333	1.0	0.9655
0.75	0.9474	0.9333	1.0	0.9655
1.0	0.9474	0.9333	1.0	0.9655

Table 14: Performance metrics of GPT-3.5 Turbo at various threshold levels on the Amazon dataset.

As observed, compared to the results from the application of other algorithms, the implementation of GPT-3.5 Turbo achieves significantly better performance metrics. The model exhibits a consistent high accuracy, preci-

sion, recall, and F1 score across varying threshold levels, indicating its robustness and reliability in categorizing products based on user reviews.

The advanced performance of GPT-3.5 Turbo can be attributed to its advanced natural language processing capabilities, which allow it to comprehend and analyze complex and nuanced user reviews effectively. This highlights the potential of integrating sophisticated LLMs like GPT-3.5 Turbo in market surveillance and other applications requiring detailed textual analysis.

The above results are further supported by the following diagrams 34, 35 depicting the Precision-Recall and ROC curves:

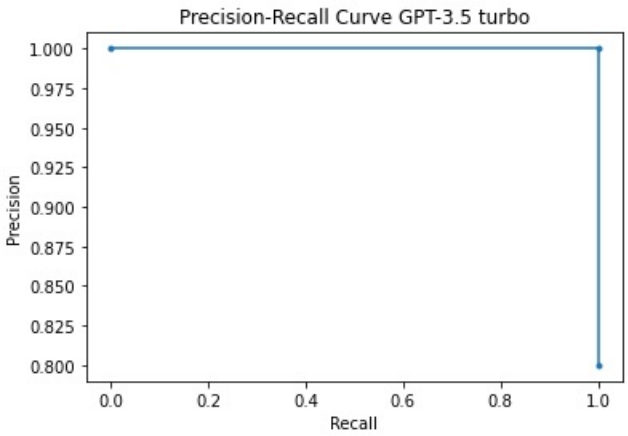


Figure 34: Precision-Recall Curve

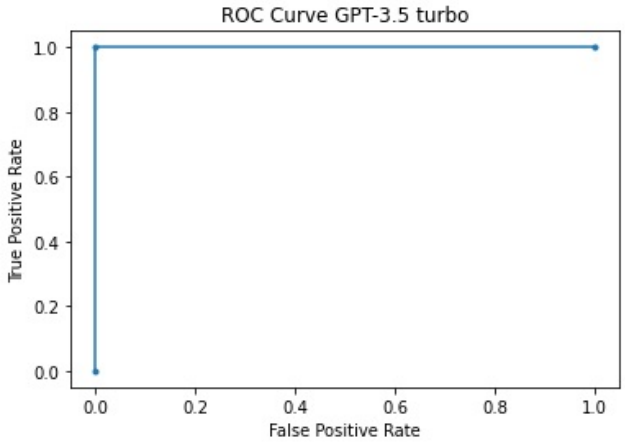


Figure 35: ROC Curve GPT3.5 turbo

5.3 Variation of Algorithm 4 Using Retrieval-Augmented Generation (RAG)

This section presents a variation of the algorithm for classifying products based on user reviews using large language models (LLMs) by incorporating the Retrieval-Augmented Generation (RAG) technique. The RAG technique enhances the model's ability to understand and analyze user reviews by integrating additional contextual information before processing. This approach improves the accuracy and reliability of the classifications by enriching the review content with relevant information. The following subsections provide a detailed explanation of the steps involved in this variation, including preparing data and API infrastructure, retrieving additional information, processing each review with RAG, aggregating outputs, and returning the final output.

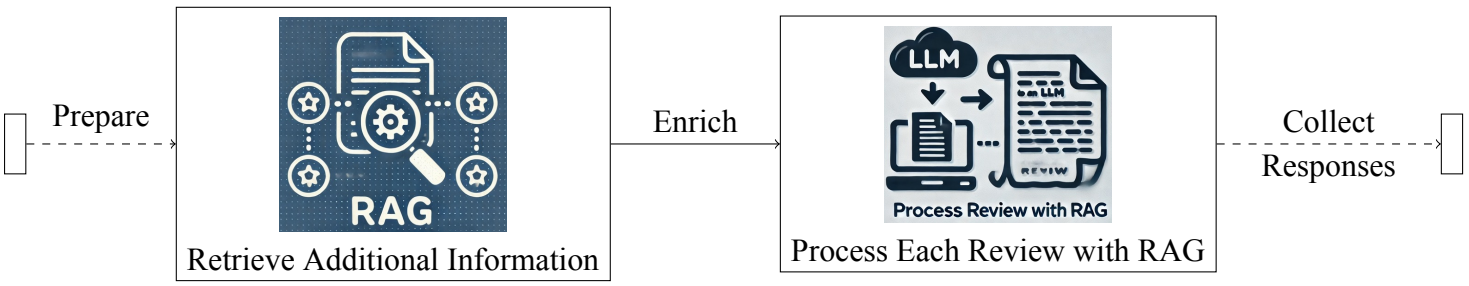


Figure 36: Visual Representation of the RAG Technique Variation for the Algorithm

5.3.1 Detailed Description of the Variation Using RAG

The following section details the modifications introduced in the variation of Algorithm 4 that utilizes the Retrieval-Augmented Generation (RAG) technique. This variation aims to enhance the LLM's ability to classify products based on user reviews by incorporating additional relevant information.

- H: Headers for the API request.
- D: Data for the API request, including the user query and reviews.
- Info: Additional contextual information retrieved for each review.
- P: Prompt created by combining the user query Q with the enriched review r + Info.
- A: Response received from the LLM API after sending the prompt.
- O: Output processed from the response A.
- O_{final}: Final aggregated output based on the user query.
- R: List of user reviews.
- Q: User query defining the desired output.
- r: Individual user review from the list R.
- I_r: Additional contextual information for the review r.
- R_r: Enriched review combining r with I_r.

Step 2: Retrieving Additional Information

Before creating the prompt for the LLM, additional contextual information is retrieved and appended to each review. This enriched review is then used to create the prompt.

$$\text{Info} = \text{Retrieve Additional Info}(r) \tag{43}$$

$$P = \text{Create Prompt}(Q, r + \text{Info}) \tag{44}$$

where: - Info is the additional contextual information retrieved for the review r. - r + Info represents the enriched review content.

Step 3: Processing Each Review with RAG

The enriched review r + Info is processed by sending the prompt to the LLM API and receiving the response.

$$A = \text{Send Prompt and Receive Response}(P, H, D) \tag{45}$$

$$O = \text{Process Output}(A) \tag{46}$$

5.3.2 Pseudocode for Algorithm 5

Algorithm 5 Classifying Products Based on User Reviews Using Large Language Models with RAG Technique

Input:
  R: List of user reviews.
  Q: User query defining the desired output.
Output:
  O: Output based on user query.
Procedure:
1. Preparing Data and API Infrastructure:
  Set up the necessary infrastructure for accessing the LLM API, including building headers and data structures.
  Headers: Define content type and authorization if needed.
  Data: Structure the user query and reviews for LLM processing.
  $H \leftarrow$ Build Headers()
  $D \leftarrow$ Build Data(Q, R)
2. Retrieving Additional Information:
  For each review r in R, retrieve additional contextual information.
  $I_r \leftarrow$ Retrieve Additional Info(r)
3. Processing Each Review with RAG:
for each review r in R **do**
    Combine the review r with additional information I_r.
    $R_r \leftarrow r + I_r$
    $P \leftarrow$ Create Prompt(Q, R_r)
    $A \leftarrow$ Send Prompt and Receive Response(P, H, D)
    $O \leftarrow$ Process Output(A)
end for
4. Aggregating Outputs:
  Combine individual outputs O into a final result based on the user query.
    $O_{\text{final}} = \text{Aggregate Outputs}(O)$
5. Returning Output:
  Return the aggregated output O.
    Return O_{final}

5.3.3 Application of GPT-3.5 Turbo with Retrieval-Augmented Generation on Amazon Dataset

In another experimentation, it is explored the use of GPT-3.5 Turbo to analyze and retrieve relevant reviews from the Amazon dataset based on specific search terms provided by the user, e.g. "bad product." This experiment serves as a guideline for future advancements in applying LLMs to extract meaningful conclusions from large text corpora.

The goal of this experiment is to demonstrate the effectiveness of the Retrieval-Augmented Generation (RAG) approach in retrieving and analyzing relevant product reviews. The methodology involves using TF-IDF and cosine similarity to retrieve relevant reviews, creating and sending requests to the GPT-3.5 Turbo API, and visualizing the results.

The performance of the model was assessed based on its ability to accurately identify reviews relevant to the given search term. For this reason, a textwrap visualization 37 was created that consolidates all the relevant reviews. This visualization helps in understanding the quality and relevance of the information retrieved by the model. Each review is displayed along with a short explanation of why it is related to the search term, providing clear insights into the model's retrieval capabilities.

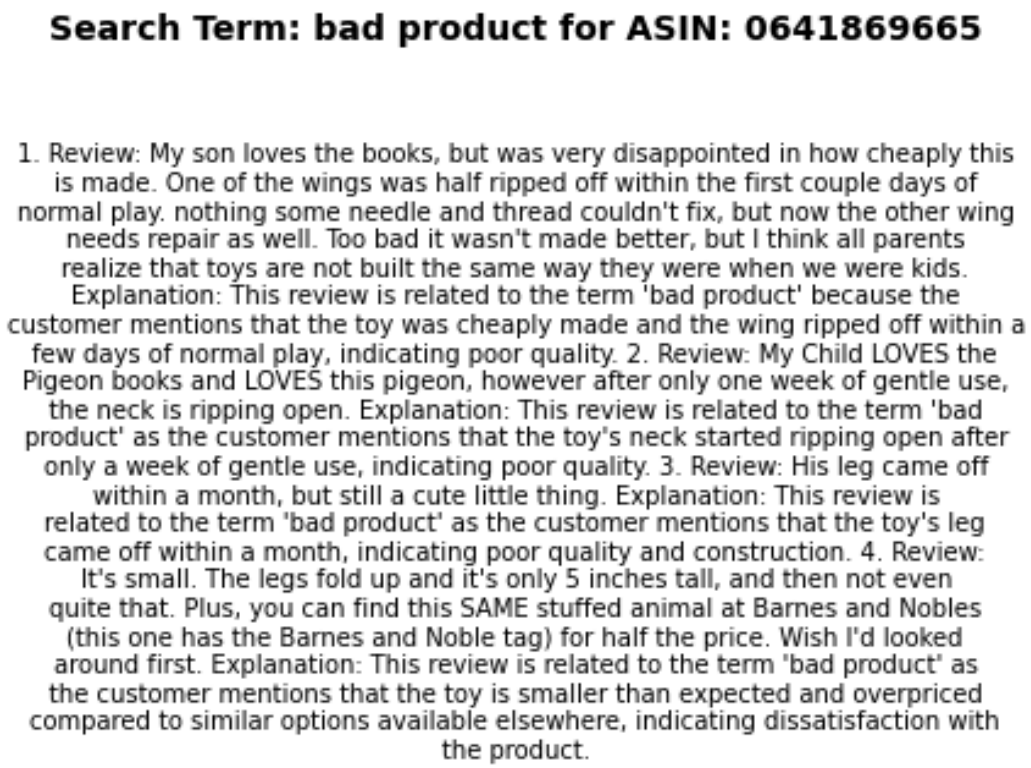


Figure 37: Textwrap Visualization of Relevant Reviews for the Search Term using RAG

5.3.4 Integration of Advanced LLMs

The application of GPT-3.5 Turbo demonstrated its superior performance in categorizing products based on user reviews. Future work should focus on integrating advanced Large Language Models (LLMs) into market surveillance systems. This integration can enhance the ability to analyze and categorize vast amounts of user-generated content more effectively, identifying potential safety issues and improving product recommendations.

Another expansion can be implemented by testing additional search terms and refining the retrieval techniques to improve the relevance and accuracy of the results. Additionally, applying this approach to other datasets and domains can help validate its effectiveness and versatility.

5.4 Challenges and Limitations

While the advanced algorithm performed well on the Amazon dataset, it is important to note the challenges and limitations encountered. The uniform nature of the synthetic dataset limited the diversity of the results, highlighting the need for more complex and varied datasets in future experiments. Moreover, the high performance of the advanced algorithm on the Amazon dataset indicates its potential, but also underscores the importance of testing in different contexts to ensure generalizability.

In conclusion, this research has demonstrated the effectiveness of semantic similarity measures and advanced LLMs like GPT-3.5 Turbo in categorizing and analyzing product reviews. The integration of these advanced techniques into market surveillance systems holds significant promise for improving product safety and user experience. Future work should focus on expanding these methods to other domains, such as Big Data and refining the algorithms to enhance their accuracy and relevance.

Bibliography

References

[1] A. Badnjević, L. Gurbeta Pokvić, A. Deumić, and L. S. Bećirović, "Post-market surveillance of medical devices: A review," *Technology and Health Care*, vol. 30, no. 6, pp. 1315–1329, 2022, doi: 10.3233/THC-220284.

[2] C. C. Freifeld, J. S. Brownstein, C. M. Menone, et al., "Digital Drug Safety Surveillance: Monitoring Pharmaceutical Products in Twitter," *Drug Safety*, vol. 37, pp. 343–350, 2014, doi: 10.1007/s40264-014-0155-x.

[3] European Union, "Directive 2009/48/EC on the safety of toys," Official Journal of the European Union, 30 June 2009. [Online]. Available: `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02009L0048-20181126`. [Accessed: 15-Jul-2024].

[4] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea, and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 8, pp. 1138–1150, 2006, doi: 10.1109/TKDE.2006.130.

[5] J. M. de Rezende, I. M. da Costa Rodrigues, L. C. Resendo, and K. S. Komati, "Combining Natural Language Processing Techniques and Algorithms LSA, Word2vec and WMD for Technological Forecasting and Similarity Analysis in Patent Documents," *Technology Analysis Strategic Management*, vol. 36, pp. 1695–1716, 2022, doi: 10.1080/09537325.2022.2110054.

[6] Z. Hajnal, "Current challenges of European market surveillance regarding products sold online," *Public Goods Govern.*, vol. 5, pp. 1–8, 2020, doi: 10.21868/PGnG.2020.1.2.

[7] J. Q. Frota Neto and M. Dutordoir, Mapping the market for remanufacturing: An application of Big Data" analytics," *International Journal of Production Economics*, vol. 230, p. 107807, 2020, doi: 10.1016/j.ijpe.2020.107807.

[8] Q. Xu, J. Li, M. Cai, and T. K. Mackey, "Use of Machine Learning to Detect Wildlife Product Promotion and Sales on Twitter," *Frontiers in Big Data*, vol. 2, p. 28, 2019, doi: 10.3389/fdata.2019.00028.

[9] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artificial Intelligence Review*, vol. 55, pp. 5731–5780, 2022, doi: 10.1007/s10462-022-10144-1.

[10] S. N. Ahmad and M. Laroche, "Extracting marketing information from product reviews: a comparative study of latent semantic analysis and probabilistic latent semantic analysis," *Journal of Marketing Analytics*, vol. 11, pp. 662–676, 2023, doi: 10.1057/s41270-023-00218-6.

[11] J. M. Gerken and M. G. Moehrle, "A new instrument for technology monitoring: novelty in patents measured by semantic patent analysis," *Scientometrics*, vol. 91, pp. 645–670, 2012, doi: 10.1007/s11192-012-0635-7.

[12] D. M. Goldberg, S. Khan, N. Zaman, R. J. Gruss, and A. S. Abrahams, "Text mining approaches for postmarket food safety surveillance using online media," *Risk Analysis*, vol. 42, no. 8, pp. 1749–1768, 2022, doi: 10.1111/risa.13651.

[13] J. Liu and O. Toubia, "A Semantic Approach for Estimating Consumer Content Preferences from Online Search Queries," *Marketing Science*, vol. 37, no. 6, pp. 930–952, 2018, doi: 10.1287/mksc.2018.1112.

[14] S. Visweswaran, J. B. Colditz, P. O'Halloran, et al., "Machine learning classifiers for Twitter surveillance of vaping: comparative machine learning study," *J Med Internet Res*, vol. 22, no. 8, art. e17478, 2020, doi: 10.3233/THC-220284.

[15] I. Atoum, "A novel framework for measuring software quality-in-use based on semantic similarity and sentiment analysis of software reviews," *J King Saud Univ Comput Inf Sci*, vol. 32, no. 1, pp. 113–125, 2020, doi: 10.1016/j.jksuci.2018.04.012.

[16] Y. Takano and Y. Kajikawa, "Extracting commercialization opportunities of the Internet of Things: Measuring text similarity between papers and patents," *Technological Forecasting and Social Change*, vol. 138, pp. 45-68, 2019, doi: 10.1016/j.techfore.2018.08.008.

[17] M. J. Hussain, S. H. Wasti, G. Huang, L. Wei, Y. Jiang, and Y. Tang, "An approach for measuring semantic similarity between Wikipedia concepts using multiple inheritances," *Information Processing & Management*, vol. 57, no. 3, 2020, doi: 10.1016/j.ipm.2019.102188.

[18] D. S. Hain, R. Jurowetzki, T. Buchmann, and P. Wolf, "A text-embedding-based approach to measuring patent-to-patent technological similarity," *Technological Forecasting and Social Change*, vol. 177, 2022, doi: 10.1016/j.techfore.2022.121559.

[19] T. S. Kim and S. Y. Sohn, "Machine-learning-based deep semantic analysis approach for forecasting new technology convergence," *Technological Forecasting and Social Change*, vol. 157, 2020, doi: 10.1016/j.techfore.2020.120095.

[20] S. Venkatraman, B. Surendiran, and P. A. R. Kumar, "Spam e-mail classification for the Internet of Things environment using semantic similarity approach," *J Supercomput*, vol. 76, pp. 756-776, 2020, doi: 10.1007/s11227-019-02913-7.

[21] M. Riyahi and M. K. Sohrabi, "Providing effective recommendations in discussion groups using a new hybrid recommender system based on implicit ratings and semantic similarity," *Electronic Commerce Research and Applications*, vol. 40, 2020, doi: 10.1016/j.elerap.2020.100938.

[22] Á. Hernández-Castañeda, R. A. García-Hernández, Y. Ledeneva, and C. E. Millán-Hernández, "Extractive Automatic Text Summarization Based on Lexical-Semantic Keywords," *IEEE Access*, vol. 8, pp. 49896-49907, 2020, doi: 10.1109/ACCESS.2020.2980226.

[23] Wu, Zhibiao, and Martha Palmer. *Verb semantics and lexical selection.* arXiv preprint cmp-lg/9406033 (1994). 10.48550/arXiv.cmp-lg/9406033.

[24] Resnik, Philip. *Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language.* Journal of artificial intelligence research 11 (1999): 95-130. 10.1613/jair.514.

[25] Jiang, Jay J., and David W. Conrath. *Semantic similarity based on corpus statistics and lexical taxonomy.* arXiv preprint cmp-lg/9709008 (1997). 10.48550/arXiv.cmp-lg/9709008.

[26] Leacock, Claudia, Martin Chodorow, and George A. Miller. *Using corpus statistics and WordNet relations for sense identification.* Computational Linguistics 24, no. 1 (1998): 147-165. Available from `https://aclanthology.org/J98-1006.pdf`.

[27] Lin, Yung-Shen, Jung-Yi Jiang, and Shie-Jue Lee. *A similarity measure for text classification and clustering.* IEEE transactions on knowledge and data engineering 26, no. 7 (2013): 1575-1590. 10.1109/TKDE.2013.19.

[28] Pedersen, Ted, Siddharth Patwardhan, and Jason Michelizzi. *WordNet:: Similarity-Measuring the Relatedness of Concepts.* In AAAI, vol. 4, pp. 25-29. 2004. Available from https://cdn.aaai.org/AAAI/2004/AAAI04-160.pdf.

[29] Sun, Yizhou, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. *Pathsim: Meta path-based top-k similarity search in heterogeneous information networks.* Proceedings of the VLDB Endowment 4, no. 11 (2011): 992-1003. 10.14778/3402707.3402736.

[30] Huang, Anna. *Similarity measures for text document clustering.* In Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand, vol. 4, pp. 9-56. 2008. Available from https://www.academia.edu/download/44422710/SMTP.pdf.

[31] Nguyen, Hieu V., and Li Bai. *Cosine similarity metric learning for face verification.* In Asian conference on computer vision, pp. 709-720. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. 10.1007/978-3-642-19309-5_55.

[32] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv preprint arXiv:1810.04805, 2018. Available from https://arxiv.org/abs/1810.04805.

[33] Hoang, Mickel, Oskar Alija Bihorac, and Jacobo Rouces. *Aspect-based sentiment analysis using BERT.* In Proceedings of the 22nd Nordic Conference on Computational Linguistics, pp. 187-196. 2019. Available from https://aclanthology.org/W19-6120.

[34] Li, Xin, Lidong Bing, Wenxuan Zhang, and Wai Lam. *Exploiting BERT for end-to-end aspect-based sentiment analysis.* arXiv preprint arXiv:1910.00883 (2019). Available from https://doi.org/10.48550/arXiv.1910.00883.

[35] Xu, Hu, Bing Liu, Lei Shu, and Philip S. Yu. *BERT post-training for review reading comprehension and aspect-based sentiment analysis.* arXiv preprint arXiv:1904.02232 (2019). Available from https://doi.org/10.48550/arXiv.1904.02232.

[36] Bello, Abayomi, Sin-Chun Ng, and Man-Fai Leung. *A BERT framework to sentiment analysis of tweets.* Sensors 23, no. 1 (2023): 506. Available from https://doi.org/10.3390/s23010506.

[37] Alaparthi, Shivaji, and Manit Mishra. *BERT: A sentiment analysis odyssey.* Journal of Marketing Analytics 9, no. 2 (2021): 118-126. Available from https://doi.org/10.1057/s41270-021-00109-8.

[38] Prottasha, Nusrat Jahan, Abdullah As Sami, Md Kowsher, Saydul Akbar Murad, Anupam Kumar Bairagi, Mehedi Masud, and Mohammed Baz. *Transfer learning for sentiment analysis using BERT based supervised fine-tuning.* Sensors 22, no. 11 (2022): 4157. Available from https://doi.org/10.3390/s22114157.

[39] Hutto, Clayton, and Eric Gilbert. *Vader: A parsimonious rule-based model for sentiment analysis of social media text.* In Proceedings of the international AAAI conference on web and social media, vol. 8, no. 1, pp. 216-225. 2014.10.1609/icwsm.v8i1.14550.

[40] Isnan, Mahmud, Gregorius Natanael Elwirehardja, and Bens Pardamean. *Sentiment analysis for TikTok review using VADER sentiment and SVM model.* Procedia Computer Science 227 (2023): 168-175. doi: 10.1016/j.procs.2023.10.514.

[41] Turner, Jason, Mehmed Kantardzic, and Rachel Vickers-Smith. *Infodemiological examination of personal and commercial tweets about cannabidiol: term and sentiment analysis.* Journal of Medical Internet Research 23, no. 12 (2021): e27307. doi: 10.2196/27307.

[42] Molenaar, Annika, Dickson Lukose, Linda Brennan, Eva L. Jenkins, and Tracy A. McCaffrey. *Using Natural Language Processing to Explore Social Media Opinions on Food Security: Sentiment Analysis and Topic Modeling Study.* Journal of Medical Internet Research 26 (2024): e47826. doi: 10.2196/47826.

[43] Deveikyte, Justina, Helyette Geman, Carlo Piccari, and Alessandro Provetti. *A sentiment analysis approach to the prediction of market volatility.* Frontiers in Artificial Intelligence 5 (2022): 836809. doi: 10.3389/frai.2022.836809.

[44] Elbagir, Shihab, and Jing Yang. *Twitter sentiment analysis using natural language toolkit and VADER sentiment.* In Proceedings of the international multiconference of engineers and computer scientists, vol. 122, no. 16. sn, 2019. Available from `https://www.iaeng.org/publication/IMECS2019/IMECS2019_pp12-16.pdf`.

[45] Ni, Jianmo, Jiacheng Li, and Julian McAuley. *Justifying recommendations using distantly-labeled reviews and fine-grained aspects.* In Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp. 188-197. 2019, Hong Kong, China. Association for Computational Linguistics. 10.18653/v1/D19-1018.
Available from: `https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/`

Appendix

Sample of Synthetic Dataset

The complete synthetic dataset is available on GitHub at the following link: `https://lmy.de/jjdwj`

```
Example dataset of tweets:
- "As a concerned parent, I appreciate the strict safety regulations that Toy1 adheres to.
It gives me peace of mind knowing my child is playing with a safe toy. #Parenting #
    ↪ ToySafety"
- "Attention parents! Toy2 has been recalled due to a potential choking hazard. Stay
    ↪ informed and
keep your children safe. #ToyRecall #SafetyFirst"
- "Beware of Toy3's excessive noise levels. Prolonged exposure can harm children's hearing.
Let's prioritize their well-being and demand quieter toys. #NoiseHazard #ChildrensHealth"
- "Shocked to find out that Toy4 has small parts that pose a choking hazard. Let's hold toy
    ↪  companies accountable for ensuring child safety. #ChildProtection #ToyManufacturers
    ↪ "
- "Parents bad quality and high risk! Be careful of Toy1's easily breakable parts. They can
    ↪  become sharp objects, endangering our children. Choose toys built to last and
    ↪ prioritize safety. #BreakageRisk #ChildSafety"
- "Shocked to discover that Toy2 contains small parts that can easily be swallowed. This
    ↪ violates EU safety regulations. Protect our children! #ChokingRisk #ChildSafety"
- "Toy3's small magnets are a serious ingestion hazard. Manufacturers must ensure secure
    ↪ closures to prevent life-threatening accidents. #MagnetIngestion #ChildProtection"
- "Disappointed with the lack of durability in Toy1. It fell apart within days, and that's
    ↪ not what I expect from a toy marketed as safe. #QualityIssues #CustomerExperience"
- "Toy2's sharp edges are a serious concern. Children's toys should be designed with
    ↪ rounded edges to prevent injuries. #ChildSafety #DesignFlaws"
- "Toy2's packaging poses a suffocation risk. Let's advocate for child-safe packaging that
    ↪ eliminates hazards and prioritizes their well-being. #PackagingSafety #
    ↪ ChildProtection"
- "I regret purchasing Toy1. Its poor design and lack of safety features make it a
    ↪ potential danger to children. Spread the word to protect others! #UnsafeDesign #
    ↪ ToyFail"
- "Toy3 passed rigorous safety tests with flying colors. It's a relief to know our little
    ↪ ones can enjoy hours of fun without any worries. #ChildSafety #QualityToys"
- "Toy4's misleading age recommendation puts younger children at risk. Manufacturers should
    ↪  provide accurate guidelines to prevent accidents. #AgeAppropriateToys #ChildSafety"
- "Disappointed with Toy4's inadequate safety labeling. It's essential for parents to have
    ↪ clear information about potential hazards. Transparency is key! #LabelingIssues #
    ↪ ConsumerSafety"
- "Warning: Toy1 poses a strangulation risk due to long cords. Parents, please be cautious
    ↪ and ensure your children's safety. #ChokingHazard #UnsafeToys"
- "Just purchased Toy1 for my niece, and I'm relieved it meets all EU safety standards! #
    ↪ SafeToys #EURegulations"
```

Function implementing Weighted Average Semantic Similarity Calculation

```
def calculate_weighted_average_semantic_similarity(review, danger_keywords, method,
    ↪ brown_ic):
    total_weight = 0.0
    total_sem_sim = 0.0

    lemmatizer = WordNetLemmatizer()
    lemmas = set()

    for lemma in danger_keywords:
        lemma_words = lemma.split()
        lemmas.update(lemma_words)

    danger_keywords = [lemmatizer.lemmatize(lemma, get_wordnet_pos(lemma)) for lemma in
        ↪ lemmas]
    danger_keywords = list(set(danger_keywords))

    raw_scores = []
    for lemma in review:
        max_sem_sim = 0.0
        for danger_keyword in danger_keywords:
            sem_sim = calculate_semantic_similarity(lemma, danger_keyword, method=method)
            if sem_sim is not None and sem_sim > max_sem_sim:
                max_sem_sim = sem_sim
        raw_scores.append(max_sem_sim)

    if method in ["res", "jcn", "lin", "lch"]:
        normalized_scores = normalize_scores(raw_scores)
    else:
        normalized_scores = raw_scores
    for score in normalized_scores:
        weight = 0.0
        if score >= 0.9:
            weight = 0.1
        elif score >= 0.6:
            weight = 0.3
        elif score >= 0.4:
            weight = 0.5
        elif score >= 0.2:
            weight = 0.8
        else:
            weight = 1.0
        total_sem_sim += score * weight
        total_weight += weight

    if total_weight != 0.0:
        return total_sem_sim / total_weight
    else:
        return 0
```