electronics

MDPI

Article

Optimising Text Classification in Social Networks via Deep Learning-Based Dimensionality Reduction

Jose A. Diaz-Garcia [1,2,*,†], Andrea Morales-Garzón [1,2,*,†], Karel Gutiérrez-Batista [1] and Maria J. Martin-Bautista [1,2]

[1] Department of Computer Science and Artificial Intelligence, University of Granada, C. Periodista Daniel Saucedo Aranda, 18014 Granada, Spain; karel@decsai.ugr.es (K.G.-B.); mbautis@decsai.ugr.es (M.J.M.-B.)
[2] Research Centre for Information and Communications Technologies, C. Periodista Rafael Gómez Montero, 18014 Granada, Spain
* Correspondence: jagarcia@decsai.ugr.es (J.A.D.-G.); amoralesg@decsai.ugr.es (A.M.-G.)
† These authors contributed equally to this work and are listed in alphabetical order.

Abstract

Text classification is essential for handling the large volume of user-generated textual content in social networks. Nowadays, dense word representation techniques, especially those yielded by large language models, capture rich semantic and contextual information from text that is useful for classification tasks, but generates high-dimensional vectors that hinder the efficiency and scalability of the classification algorithms. Despite this, limited research has explored effective dimensionality reduction techniques to balance representation quality with computational demands. This study presents a deep learning-based framework for enhancing text classification in social networks, focusing on computational performance, by compressing high-dimensional text representations into a low-dimensional space while retaining essential features for text classification. To demonstrate the feasibility of the proposal, we conduct a benchmarking study using traditional dimensionality reduction techniques on two widely used benchmark datasets. The findings reveal that our approach can substantially improve the efficiency of text classification in social networks without compromising—and, in some cases, enhancing—the predictive performance.

Keywords: dimensionality reduction; deep learning; social media mining; text classification

Academic Editor: Kefeng Ji

Received: 5 June 2025
Revised: 30 July 2025
Accepted: 21 August 2025
Published: 27 August 2025

Citation: Diaz-Garcia, J.A.; Morales-Garzón, A.; Gutiérrez-Batista, K.; Martin-Bautista, M.J. Optimising Text Classification in Social Networks via Deep Learning-Based Dimensionality Reduction. *Electronics* **2025**, *14*, 3426. https://doi.org/10.3390/electronics14173426

1. Introduction

The exponential growth of user-generated textual content on social network platforms has intensified the demand for efficient and accurate text classification methodologies. Tasks like sentiment analysis [1–3], topic detection [4,5], and spam filtering [6–8] are vital for enhancing user experience and ensuring content relevance [9]. Most of these works focus on developing a methodology or training a model to enhance classification accuracy, while overlooking the efficiency of the proposal.

Dimensionality reduction techniques are crucial for improving text classification performance by reducing computational complexity and, in some cases, enhancing accuracy. Feature selection methods, such as Chi-Square and Information Gain Ratio, and feature extraction techniques like Principal Component Analysis (PCA) and Latent Semantic Analysis (LSA), have proven to be effective [10]. Studies have demonstrated that dimensionality reduction often improves classification accuracy compared to non-reduced data [11]. Some researchers have proposed approaches such as combining clustering with Hidden Markov Models [12] or leveraging taxonomic information [13]. While dimensionality reduction

generally improves preprocessing and training times, there are exceptions where significant accuracy losses can occur [14].

In recent years, deep learning (DL) models, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transformer architectures, have demonstrated remarkable success in capturing complex patterns within textual data, thereby achieving state-of-the-art results in many downstream natural language processing (NLP) tasks [15]. These models, as classical ones, often rely on high-dimensional word embeddings, such as Word2Vec [16], GloVe [17], and contextual embeddings from models like BERT [18], to represent textual information.

While these dense representations encapsulate rich semantic and syntactic nuances, their high dimensionality poses significant challenges. Specifically, they can lead to increased computational costs, longer training times, and difficulties in deploying models on resource-constrained devices. These limitations not only affect scalability and accessibility, but also raise concerns about computational efficiency, energy consumption, and consequently environmental impact. The latter has garnered attention in recent years, particularly in the context of green computing and the Sustainable Development Goals (SDGs), such as those related to responsible consumption and climate action. Moreover, high-dimensional data can exacerbate the curse of dimensionality, potentially affecting model generalisation and performance [19]; that is why these types of algorithms also need to reduce dimensionality to improve performance from an efficiency perspective.

More recently, DL-based approaches, such as autoencoders (AEs), have been widely used for dimensionality reduction due to their ability to learn non-linear transformations and capture intricate data structures [20,21]. AEs are straightforward unsupervised neural networks that can obtain a better representation of the original data in a low-dimensional space. As traditional techniques for dimensionality reduction, most studies using AEs focus on enhancing classification accuracy or improving algorithm complexity.

Despite these advancements, a lack of research persists with regard to in-depth analysis of how dimensionality reduction affects text classification in social network contexts. Taking into account the special characteristics of social media textual data, such as informal language, abbreviations, and the rapid evolution of topics, it is crucial to effectively reduce dimensionality and obtain the best possible configuration without compromising classification performance [22,23].

This study proposes an efficient and effective framework based on DL that can optimise text classification tasks in social networks by addressing the intersection of dimensionality reduction and text classification. Our approach systematically evaluates the impact of dimensionality reduction on classification accuracy and computational efficiency across multiple setups on different benchmark datasets. Through experimentation and analysis, we aim to identify configurations that alleviate computational constraints while maintaining, or potentially enhancing, the accuracy of text classification models. Based on this, and to support our hypothesis that dimensionality reduction techniques can improve the efficiency of text classification without reducing accuracy, and even improving it, this paper addresses the following research questions:

- RQ1: Can dimensionality reduction techniques improve text classification performance in terms of accuracy, processing time, and carbon emissions?
- RQ2: Are DL-based dimensionality reduction frameworks effective and worthwhile compared to traditional methods?
- RQ3: Do dimensionality reduction techniques provide a significant improvement in processing time while maintaining comparable accuracy levels?
- RQ4: Are all original features necessary to preserve classification accuracy, or can similar performance be achieved with fewer dimensions and reduced model complexity?

It is important to note that the goal of this study is not to identify the optimal number of features or the single best dimensionality reduction technique. Rather, our aim is to provide a robust comparative analysis of the impact of various dimensionality reduction methods on text classification performance. In the course of conducting this research and addressing the proposed research questions, the paper makes the following contributions:

1. The development of a robust and flexible DL-based framework for improving text classification in social networks from a computational (time and carbon emissions) perspective without affecting, and even improving, the accuracy of the classification.
2. A thorough evaluation of the proposed DL-based framework. We assess the framework using two benchmark datasets, employing different dimensions, several machine learning algorithms, and traditional and DL-based dimensionality reduction techniques. We also provide an ablation study considering different components included in the proposal.
3. An in-depth analysis of the trade-offs between dimensionality reduction and classification performance, providing insights into different configurations, classification times, and carbon emissions.

The rest of the paper is organised as follows: Section 2 presents the most important related work in the literature. Section 3 describes the framework for improving text classification performance through dimensionality reduction. Section 4 describes the experiments we conducted, Section 5 presents the results of the ablation study, and Section 6 provides a discussion of the obtained results. Section 7 outlines final considerations and future research directions.

2. Related Work

As stated, dimensionality reduction is a crucial phase in text classification tasks. It is not an easy problem due to the high dimensionality of the data. Various approaches have been developed to address the curse of dimensionality, ranging from feature selection and feature extraction to clustering-based and DL-based methods. Each group employs different tools and techniques to deal with high-dimensional data.

The paper presented in [24] proposes a dimensionality reduction method called removal of redundant features (rRF) that uses the GloVe [17] word embedding technique to identify and remove redundant features in text classification and also introduces a new performance metric called the New Performance Metric (NPM) to evaluate the classifier's performance on the reduced feature set. In [25], the authors propose a novel approach for aggressive dimensionality reduction in text categorisation using local feature selection, global weighting scaling, and feature selection measure enhancement.

In [26], the author compares three dimensionality reduction algorithms (PCA, Non-negative Matrix Factorization (NMF), and Singular Value Decomposition (SVD)) for text clustering and finds that PCA gives the best results in terms of clustering quality and interpretability for both Arabic and English documents. The work presented in [27] investigates the influence of dimensionality reduction on the subject classification of text documents in Polish, finding that PCA outperforms random projection and that FastText [28] can be significantly reduced in dimensionality without losing quality.

In [29], the authors propose an efficient ontology-based approach for dimensionality reduction in web text document classification that outperforms traditional PCA methods. The approach uses WordNet ontology and its hierarchical structure to eliminate words from the generated feature vector that have no relation to any of WordNet's lexical categories. In compliance with [30], creating variables to reduce the dimensionality of text categorisation tasks using a Self-Organising Map (SOM) performs better than random mapping and is an excellent alternative to PCA.

A comprehensive analysis of dimensionality reduction techniques is presented in [31]. The authors evaluate different dimensionality reduction techniques for effective text classification, recommending feature selection and feature transformation, followed by feature selection, as well as the use of symbolic interval data types. They evaluate these techniques using Support Vector Machines (SVMs) and symbolic classifiers on standard datasets. In [32], the authors empirically demonstrate the performance increase of active learning, using a K-Nearest-Neighbour classifier, when dimensionality reduction is applied using unsupervised techniques, such as document frequency and Principal Component Analysis.

Another interesting approach is presented in [33], where a new dimensionality reduction method for text classification is proposed by applying the discrete wavelet transform to the document-term frequency matrix. In [34], the authors evaluate a scatter method for dimensionality reduction in text classification. According to [35], utilising clustering and Hidden Markov Models for dimensionality reduction allows for the enhancement of text classification problems on the OHSUMED dataset using K-Nearest Neighbours (K-NN) and SVM classifiers. Similarly, utilising LSI and SVD techniques to reduce the dimensionality of the document vectors in the MEDLINE dataset enables improvements in centroid-based classification and Support Vector Machine performance [36].

The use of rough set theory [37] for reducing the dimensionality of text classification features improves text classification efficiency, accuracy, and speed compared to vector space-based approaches. Dimensionality reduction and feature selection techniques, such as the Discrete Cosine Transform (DCT) and the Proportion of Variance, have proven to be more effective for classification results in text classification by reducing data size while preserving classification accuracy [38].

A concept indexing-based approach is presented in [39] to improve text classification accuracy. Additionally, the proposal enhances efficiency and scalability in experiments conducted on the Reuters 21578 dataset. Using the same dataset, in [40], the author depicts another work related to feature reduction for text classification. The algorithm defines and quantifies features, removes standard features, and adjusts the weights of multi-source features to improve classification performance. Ref. [11] introduces an approach for feature representation and dimensionality reduction of text documents, which retains the original feature distribution and produces a low-dimensional matrix. The proposal utilises techniques such as clustering and fuzzy logic, resulting in a low-dimensional matrix that is better suited for text clustering and classification.

In recent years, deep learning and transformer-based architectures have increasingly been explored for dimensionality reduction. For example, Boyapati et al. [41] propose Semanformer, a transformer-based encoder–decoder framework designed to reduce the dimensionality of BERT embeddings while preserving semantic integrity. Unlike traditional methods like PCA, t-SNE, UMAP, or SVD, which often compromise semantic fidelity, Semanformer uses autoregressive modeling to maintain high-quality sentence reconstruction, achieving over 83% reconstruction accuracy compared to less than 66% with PCA and under 9% with t-SNE. These results underscore the potential of transformer-based approaches to improve dimensionality reduction in NLP, a motivation that aligns with the goals of our study. In the biological domain, where datasets are typically high-dimensional, transformer-inspired models have also shown promise. Thor et al. [42] introduce a contrastive learning framework optimized for SNP genotype data, combining novel data augmentation with a loss function tailored for genetic population structure visualization. The model effectively captures both local and global relationships, outperforming PCA, t-SNE, and UMAP in preserving structure and generalization—demonstrating a strategy similar to transformer-based representation learning. Further, Nareklishvili et al. [43] present DETs, a deep architecture combining decision tree ensembles with transformer

design principles. Although not based on LLMs, DETs offer an interpretable and flexible alternative to PCA, showing superior performance on biomedical benchmarks. Lastly, Sakr et al. [44] introduce ESPACE, a novel activation-centric compression method for LLMs. Instead of altering weights like SVD or SliceGPT, ESPACE projects activation tensors onto principal components while preserving original weights, enabling efficient retraining. It achieves up to 50% compression with minimal perplexity loss, improves accuracy at lower compression rates, and reduces inference latency by up to 45%.

Generally speaking, previous studies have only focused on applying dimensionality reduction methods in order to improve the classification accuracy of models. In this work, unlike previous ones, we propose a framework that focuses on finding an efficient configuration to reduce the execution time and carbon footprint of classification models, while also improving text classification performance by achieving higher accuracy through optimised dimensionality. The framework presents a pipeline that encompasses dataset-specific preprocessing, fine-tuning of transformer-based models, dimensionality reduction, and final classification. To demonstrate the robustness of our approach, we assess our framework using two widely used social media datasets, conducting a thorough analysis of its performance. By systematically evaluating the trade-offs between computational efficiency and classification performance, the proposed study provides a more complete perspective on text classification in social networks than previous works.

3. DL-Based Framework for Optimising Text Classification in Social Networks

This work proposes a DL-based system to optimize the text classification task in social networks. The proposal aims to address the performance of the classification algorithms in terms of computational burden without affecting the performance of the algorithm in terms of classification accuracy. To do so for each analysed dataset, we first preprocess the textual data. Then, we create a training dataset to fine-tune an LLM used to obtain the representation of the text in a high-dimensional vector space. Afterwards, we apply a dimensionality reduction technique in order to reduce the dimensionality of the original problem. Finally, we employ a classical classification algorithm using the features obtained in the previous step. The whole workflow is summarised in Figure 1. A detailed description of each step is given below.

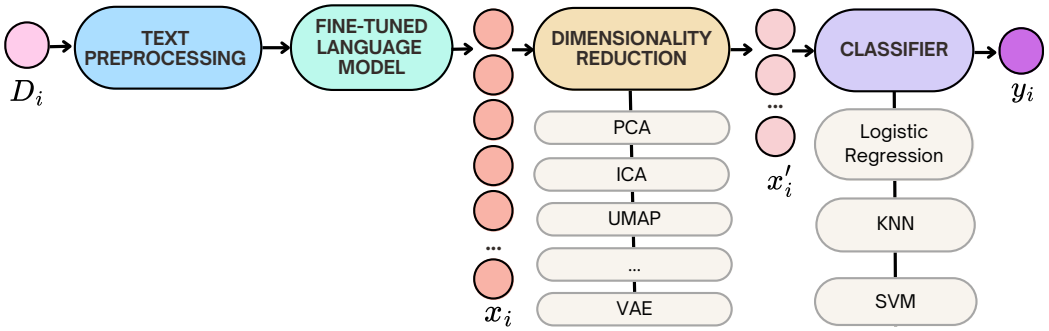


Figure 1. General overview of the proposed framework for optimising text classification in social networks.

3.1. Text Preprocessing

In this step, the text is cleaned to reduce noise while preserving key semantic elements. Specifically, we retain stop words, punctuation marks, and emoticons, as they often convey sentiment. Since we use the all-mpnet-base-v2 model, which employs a more robust SentencePiece tokenizer compared to the traditional BERT WordPiece tokenizer, emoticons are processed without being mapped to an [UNK] (unknown) token. Unlike earlier tok-

enizers, this allows the model to generate embeddings for Unicode emojis directly. While associating semantic meaning with emoticons has been a challenge in the literature [45,46], modern tokenizers now preserve their form and enable the model to interpret them within context, maintaining their contribution to the overall sentence representation.

We remove non-textual elements: all HTML tags (e.g., ' ', '
'), URLs, and any embedded metadata such as geolocation coordinates or tweet IDs. As we use two benchmark datasets, dataset-specific rules are applied: for example, we remove any formatting tags in movie reviews (from the IMDb dataset) [47], whereas on BullyingV3.0 posts (from Twitter) [48], we delete social media artifacts (retweet markers, geo-tags, etc.). This process ensures that the input to the LLM consists of clean text enriched only with linguistically meaningful content. The preprocessing pipeline can be summarised as follows:

- Retained elements: Stop words, punctuation, and emoticons (to preserve semantic and emotional content).
- Removed elements: HTML tags, embedded URLs (e.g., 'http://'), geolocation fields, and other non-linguistic metadata.
- Dataset-specific cleaning: For IMDb reviews, delete any HTML artifacts while preserving sentence structure. For BullyingV3.0 tweets, additionally, remove user mentions or retweet markers and drop coordinate tags, focusing only on the message text.

3.2. Large Language Model Fine-Tuning

Recent advancements in DL, particularly in LLMs, have significantly enhanced the performance of various downstream NLP tasks. One of the primary applications of this type of model is to obtain a feature-vector representation of textual data. This representation encompasses contextual information and important nuances related to the text, useful for several downstream NLP tasks, such as text classification.

In the state of the art, we can find different pre-trained LLMs for various purposes and in different flavours (character embeddings [28], word embeddings [16–18,49,50], and sentence embeddings [51]). Pre-trained language models are built using extensive corpora such as Wikipedia and Google datasets. While these models yield outstanding results in various downstream NLP tasks, fine-tuning them on domain-specific data can enhance their effectiveness. This process involves adjusting the model parameters in order to capture the particular characteristics of the target domain, thereby enhancing performance on specialised tasks.

Considering the aforementioned statement, we decided to fine-tune a pre-trained LM in order to obtain better results. Specifically, we used Sentence-BERT (SBERT) [51] for text representation. SBERT (based on BERT) enables the representation of sentences, paragraphs, and images through dense vectors. SBERT aims to generate meaningful sentence embeddings. SBERT provides an extensive list of pre-trained models for over 100 languages. For our approach, we utilised all-mpnet-base-v2 (https://huggingface.co/sentence-transformers/all-mpnet-base-v2 (accessed on 23 July 2025)), a model widely employed in similarity-related tasks, and we fine-tuned it on our task using contrastive learning [52], the inherent learning strategy of the model.

Contrastive learning involves training a model to differentiate between similar and dissimilar examples. The goal is to learn a representation of the input data that captures the underlying structure of the data while also ensuring that similar examples are mapped close in embedding space. In contrast, dissimilar examples are pushed apart to distant points. The model is trained using pairs of examples, where each pair consists of a positive example (two similar examples) and a negative example (two dissimilar examples). The model learns to maximise the similarity between the positive examples while minimising the similarity between the negative examples. This kind of learning has been effective in a

wide range of applications, including computer vision, natural language processing, and speech recognition. It is often used as a pre-training method to learn a representation of the data that can be fine-tuned on downstream supervised learning tasks.

Dataset for Fine-Tuning the Large Language Model

To fine-tune the chosen model, for each dataset, we create a training set where each instance consists of a pair of sentences accompanied by a numerical value representing their semantic similarity. This training set for fine-tuning is derived from the training portion of the original dataset, which is initially partitioned into training and validation subsets for classification purposes.

To perform sentence pairing, a sentence from the original dataset is paired with a sentence from the same class and one from a different class, both randomly selected, to form training pairs. Pairs with the same class are assigned a similarity score of 1, while those with different classes receive a score of 0. This process enables the model to learn semantic representations based on similarity degrees. The pairing process can be repeated across multiple iterations to expand the training set.

As previously mentioned, SBERT offers a comprehensive selection of pre-trained models. In this study, we utilise the all-mpnet-base-v2 model, which encodes sentences and paragraphs into 768-dimensional dense vectors. These embeddings are suitable for various natural language processing tasks such as clustering, semantic similarity, retrieval, classification and semantic search. After fine-tuning, the model is able to encode each text into a context-aware dense vector that captures its nuanced meaning, creating similar embeddings in the vector space for sentences of the same class while creating dissimilar embeddings for sentences of different classes.

3.3. Dimensionality Reduction

Once we obtain the sentence embeddings, we can use a dimensionality reduction technique to obtain a compressed representation of the texts. This representation in a low-dimensional space must preserve semantic nuances, enabling classification algorithms to perform well in terms of accuracy. It is worth highlighting that we can apply any dimensionality reduction technique during this step, making our approach flexible and robust, as is shown in Figure 1. For dimensionality reduction, we use both linear and non-linear approaches. The selection of these approaches is intended to offer a comprehensive overview of dimensionality reduction techniques by encompassing both well-established traditional methods and cutting-edge deep learning-based approaches. Traditional techniques, such as PCA and t-SNE, provide interpretable, mathematically grounded frameworks for reducing dimensionality, often emphasising variance preservation or local structure. In contrast, recent deep learning methods leverage neural architectures to capture complex, non-linear feature relationships and scalability to large datasets. By including this diverse set of techniques, we aim to evaluate the strengths and limitations of each approach across different data characteristics and application scenarios. The methods used are as follows:

- Linear methods: PCA [53], Independent Component Analysis (ICA) [54], Locally Linear Embedding (LLE) [55], and Truncated SVD [56] are applied to reduce dimensionality. PCA and Truncated SVD find orthogonal projections that maximise variance, ICA seeks statistically independent components, and LLE preserves local neighbourhood geometry.
- Non-linear methods: We use Uniform Manifold Approximation and Projection (UMAP) [57], a dimensionality reduction technique grounded in Riemannian geometry and algebraic topology. UMAP constructs a fuzzy topological graph where each data point is connected to its nearest neighbours via weighted edges. It then opti-

mises a low-dimensional embedding that preserves the local structure by maintaining pairwise relationships between connected points. In addition, we implement shallow autoencoder (AE) and variational autoencoder (VAE) models. An autoencoder is a type of neural network consisting of two main components: an encoder, which compresses the input into a latent representation, and a decoder, which reconstructs the original input from this representation. The VAE extends this concept by introducing probabilistic inference, aiming to model the distribution of the latent space—typically assuming a Gaussian prior—rather than producing a fixed latent vector. This makes VAEs more robust and capable of generalising beyond the training data. Both models compress data through a low-dimensional bottleneck and reconstruct the input, allowing them to serve as effective non-linear dimensionality reduction techniques. Our implementations use one or two hidden layers with ReLU activation functions and a sigmoid function in the output layer. The VAE includes stochastic latent variables with a KL-divergence regulariser. The training phase is conducted using the Adam optimiser and binary cross-entropy loss (Equation (1)) to minimise reconstruction error. Formally, for input vector x and reconstruction $\hat{x}$, the loss is:

$$L_{\text{BCE}}(x, \hat{x}) = -\sum_i [x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i)], \quad (1)$$

encouraging $\hat{x} \approx x$. Both AE and VAE use a compact latent size to ensure meaningful compression.

These methods allow us to compare simple linear projections with non-linear embeddings. In practice, we fit each model on the training data and select the reduced-dimensional representations for the next stage.

It is worth highlighting that, although the literature contains several techniques, such as quantisation, pruning, model compression, and parameter-efficient fine-tuning (PEFT), for optimising resources during text representation computation for text classification, these are outside the scope of our paper. Our contribution is to evaluate both classical and deep learning-based reduction methods (e.g., PCA, LLE, VAE, AE) as a lightweight, model-agnostic post-processing step for text embedding representations, and to provide a thorough analysis of the effects of such dimensionality reduction techniques on text classification from both resource and performance perspectives.

3.4. Text Classification

Once the dimensionality reduction technique is applied to the high-dimensional text representations, we classify the instances into their respective categories using supervised algorithms. We selected three widely used classifiers: K-Nearest Neighbours (K-NN) [58], Logistic Regression (LR) [59], and Support Vector Machines (SVMs) [60]. These models provide a balance of simplicity, interpretability, and strong performance in text classification. Moreover, they represent diverse classification paradigms—instance-based (K-NN), probabilistic (LR), and margin-based (SVM), allowing us to evaluate how dimensionality reduction impacts different classification approaches.

We recognise that adding a single fully connected layer (FCL) on top of embeddings is computationally light at inference time and may yield reasonable accuracy. Recent work shows that fine-tuning only the final layer can be much faster than fine-tuning the full encoder, since the encoder is frozen and only the last layer is trained [61].

However, our proposal focuses on a different scenario. We aim to reduce precomputed high-dimensional embeddings into a lower-dimensional space once, and then repeatedly use those reduced representations for fast text classification. While a single FCL is simple and often fast, our method proposes a classification pipeline that decouples the processes

of obtaining embeddings, compression, and classification. This enables faster repeated inference with multiple lightweight classifiers, richer analysis of embedding quality, and maximal flexibility.

4. Experimentation

This section provides detailed information to ensure the reproducibility and robustness of our approach by testing different combinations of vector size reductions (i.e., dimensions) and pre-trained and fine-tuned language models.

4.1. Data

For our experiments, we utilised two distinct datasets (Table 1) sourced from diverse social media platforms: Twitter (now X) [48] and IMDb (an online database of information related to films) [47]. This selection allows for a robust evaluation of dimensionality reduction techniques in a variety of linguistic and contextual environments. We chose these two sources as their elements have different text lengths and class distributions to ensure a strong assessment.

Table 1. Dataset description.

Name	Size	Classes	Social Platform	Class Distribution
IMDb	50,000	2	IMDb	25,000 samples per class
BullyingV3.0	7321	2	Twitter	2102 bullying samples (28.71%)

The preprocessing phase was designed to preserve the linguistic elements that contribute to semantic understanding. Stop words, punctuation marks, and emoticons were retained, as they play a potential role in conveying context and sentiment. However, non-semantic embedded elements, such as HTML tags, were removed. Specific cleaning procedures varied by dataset: for IMDb, HTML tags were stripped from textual data; for BullyingV3.0, sourced from Twitter, more extensive cleaning was necessary. The process involved the removal of embedded URLs, geolocation marks, and tags from the attached media. Table 2 presents the descriptive statistics of the cleaned textual content, measured in character length.

Table 2. Descriptive statistics datasets after the cleaning phase (measured in characters).

Statistic	IMDb	BullyingV3.0
Mean	1289.214	89
Standard Deviation	974.187	36.551
Minimum	32	5
First Quartile	691	58
Second Quartile (Median)	957	90
Third Quartile	1563	124
Maximum	13,604	144

We additionally computed word-level and BERT token-level statistics for each sample in both datasets. Table 3 presents the results for BullyingV3.0, while Table 4 shows the statistics for the IMDb dataset. For IMDb, the analysis reflects the preprocessed (i.e., truncated and merged) sequences, with a maximum input length of 384 tokens as described in Section 4.2. Tokenization was performed using the BERT base uncased tokeniser. While minor differences in token counts were observed across classes, they did not introduce

architectural bias, as BERT inputs were padded or truncated to a fixed sequence length before being passed to the model.

Table 3. Word and BERT token statistics per class in the BullyingV3.0 dataset.

Class	Mean Words	Median Words	Std. Words	Mean Tokens	Median Tokens	Std. Tokens
Non-Bullying (0)	10.65	10	5.00	20.49	18	10.35
Bullying (1)	8.87	9	4.11	15.80	16	6.04

Table 4. Word and BERT token statistics per class in the IMDb dataset.

Class	Mean Words	Median Words	Std. Words	Mean Tokens	Median Tokens	Std. Tokens
Negative (0)	225.69	171	162.66	254.14	192	183.35
Positive (1)	231.85	171	176.84	260.04	191	198.85

Regarding data split, all datasets were divided using a stratified approach to preserve class distributions. Specifically, 70% of each dataset was allocated to training and 30% to testing. Furthermore, due to the use of DL models, the training set was further divided into training and validation subsets, with 80% used for training and 20% for validation. This partitioning strategy ensures unbiased model evaluation and effective hyperparameter tuning.

4.2. Model Configuration and Parameters

Choosing the correct model configuration is crucial, as it directly affects the quality of textual representations and, consequently, the performance of downstream classification tasks. After evaluating multiple language models based on their performance on sentence-based tasks, technical characteristics, and domain adaptability, we selected all-mpnet-base-v2 https://www.sbert.net/docs/sentence_transformer/pretrained_models.html (accessed on 23 July 2025) [62], which consistently achieves state-of-the-art results in sentence embedding tasks. Although the smaller and faster all-distilroberta-v1 [63] was considered due to its efficiency, it was ultimately excluded due to its slightly lower embedding performance and large computational load.

The all-mpnet-base-v2 model is a powerful sentence encoder based on microsoft/mpnet-base [64] and fine-tuned with contrastive learning on one billion sentence pairs [62]. This training enables the model to distinguish semantic similarities effectively by focusing on relevant features and reducing noise. Its 768-dimensional embeddings capture essential information while minimising redundant features, making it well-suited for tasks requiring meaningful, compact representations like dimensionality reduction and similarity assessment. The model consistently achieves top results on benchmark datasets [65], supporting its effectiveness in diverse NLP applications.

Among the available Sentence-BERT variants, all-mpnet-base-v2 achieves state-of-the-art performance across several benchmarks in semantic textual similarity because of the quality of the embeddings, as reported on the official Sentence-BERT leaderboard. This strong behaviour makes it a strong choice for our analysis.

It is important to consider the input length limitation of the selected model, which by default truncates text exceeding 384 tokens. As shown in Table 2, this constraint poses no issue for the BullyingV3.0 dataset, as samples are well below this threshold. However, IMDb, with average lengths of 1289 characters, requires additional processing to avoid significant information loss. To address this, we implemented a sentence-level embedding strategy beyond relying on default truncation. Each text is first split into individual

sentences, from which embeddings are computed separately. These are then aggregated using mean pooling to produce a single fixed-size vector representation for the full text. This approach preserves the model's dimensional structure while allowing for the effective representation of longer inputs. After obtaining the textual representations from the pre-trained SentenceTransformers models, fine-tuning was conducted to optimise performance on each specific dataset.

The model uses ReLU (Rectified Linear Unit) as the activation function for all layers except the final layer, which uses a Sigmoid activation. Training was performed using the Adam optimiser with a learning rate of 0.0001 and the binary cross-entropy loss function. The models were trained for a maximum of 100 epochs, with early stopping based on validation loss to prevent overfitting.

5. Ablation Study

In this section, we present the results of our experiments through an ablation study designed to measure the influence of various components in our framework. To evaluate whether dimensionality reduction techniques can improve performance without compromising classification accuracy, we employed three different classifiers: K-NN, LR, and SVM. All models were implemented using the Scikit-learn library [66] with default parameters. To ensure reliability and reproducibility, each experiment was run with three different random seeds, and the final results represent the average across these runs. The machine used in the experiments features a common hardware configuration, including a 2.0 GHz quad-core Intel Core i5 processor (Intel Corporation, Santa Clara, CA, USA), 16 GB of 3733 MHz LPDDR4X RAM, Intel Iris Plus Graphics with 1536 MB of VRAM, and a SATA SSD disk.

The ablation study was specifically designed to assess the impact of fine-tuning together with several dimensionality reduction techniques. For each dataset, we tested two configurations: one with fine-tuning of the model and embeddings, and one with the pre-trained language model. Subsequently, to evaluate the influence of dimensionality reduction, we conducted additional experiments incorporating various dimensionality reduction techniques into our pipeline. For the IMDb dataset, which contains longer texts, we report results using the sentence-level averaging technique described in Section 4.2.

The primary goal of our experiments is to understand how dimensionality reduction techniques and fine-tuning affect the performance of different classification algorithms across multiple datasets. To this end, we evaluated the classifiers in various reduced dimensions, specifically in 5, 10, 30, 50, 100, 200, and 300 vector dimensions.

It is worth noting that our proposed pipeline always includes the text preprocessing techniques described in Section 3.1. Additionally, for each configuration, the same classification algorithms are tested. Therefore, the components analysed in our ablation study are whether fine-tuning is applied or not, and whether dimensionality reduction is used or not.

To improve clarity and readability, this section presents results in terms of classification accuracy for each dataset and configuration. Graphical results from all experimental runs are provided in Appendix A. Additionally, for datasets exhibiting class imbalance, we include the F1 score, which combines precision and recall via their harmonic mean, offering a more informative metric in such scenarios.

5.1. IMDB Dataset Results

Fine-tuning is the second component of our framework, analysed separately in the ablation study. Table 5 presents the results on the IMDb dataset, with and without fine-tuning. As the main objective of our work is to demonstrate that certain dimensionality reduction techniques can improve inference time and reduce emissions—without a decrease in classi-

fication accuracy, and in some cases even improving it—the values presented in Table 5 serve as the baseline for our experiments. This baseline corresponds to the classification performance obtained without applying any dimensionality reduction, meaning all original dimensions are used. In these baseline scenarios, Logistic Regression (LR) stands out as the most accurate model in both settings (with and without fine-tuning). The results across all experiments show that fine-tuning consistently enhances classification performance.

Regarding the text processing step of our ablation study, Tables 6 and 7 show the results of incorporating different dimensionality reduction techniques into our pipeline. For each technique, we test both fine-tuning (Table 6) and no fine-tuning (Table 7) configurations, evaluating various levels of dimensionality reduction ranging from 5 to 300 dimensions.

For the IMDb dataset, it is challenging to identify a consistent pattern in the results, as performance varies depending on the combination of the model and dimensionality reduction method. However, PCA and VAE generally tend to yield better outcomes. In this experiment, the best-performing configuration is the combination of fine-tuning with VAE reduction to 200 dimensions. This is a significant finding, as it demonstrates that our system can achieve improved classification performance even after reducing the feature space by 568 dimensions (from 768 to 200, i.e., a 73.96% reduction), while also benefiting from important reductions in computational time.

Another noteworthy result from PCA is that with only five dimensions, it achieves 92.94% accuracy using the LR classifier—surpassing the baseline fine-tuned accuracy of 92.59%. This highlights the effectiveness of dimensionality reduction techniques in preserving important information and knowledge while eliminating redundant or irrelevant features. Among the techniques evaluated, ICA, LLE and UMAP consistently showed the poorest performance, both with and without fine-tuning. To evaluate if the differences were meaningful, we conducted Wilcoxon [67] signed-rank tests across all classifiers and training conditions. VAE significantly outperformed PCA with SVM under fine-tuning ($p < 0.05$). For LR and K-NN, the differences were not statistically significant ($p = 0.5$ and $p = 0.84$, respectively), and no significant improvements were found for any model without fine-tuning (all p-values ≥ 0.92). These results must be interpreted in the context of already high baseline accuracies, often exceeding 92%, which limit the margin for improvement. In this scenario, even marginal but consistent gains are relevant, especially in scenarios where computational efficiency is important. Furthermore, the fact that no significant performance degradation is observed, especially when the feature space is drastically reduced, reinforces the conclusion that VAEs can offer computational benefits without compromising accuracy.

Table 5. Performance metrics on the IMDb dataset using default truncation and sentence embedding averaging, with and without fine-tuning. Best results are highlighted in bold.

Model	Without Fine-Tuning		With Fine-Tuning	
	Accuracy	**F1**	**Accuracy**	**F1**
K-NN	90.99	90.94	91.45	91.49
SVM	91.98	92.00	92.03	92.05
LR	**92.51**	**92.52**	**92.58**	**92.59**

Table 6. Accuracy on IMDb dataset with different dimensionality reduction techniques and without fine-tuning. Best results are highlighted in bold.

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
K-NN	Accuracy	300	**91.03**	70.94	90.99	89.74	90.16	90.80	88.33
		200	**91.06**	74.49	91.03	90.10	90.04	90.73	87.90
		100	**91.14**	79.72	91.09	90.28	90.16	90.69	89.33
		50	**91.02**	82.31	91.02	90.18	89.98	90.59	90.12
		30	**91.08**	84.36	90.98	90.17	90.38	90.69	89.64
		10	**90.43**	88.60	90.42	90.07	90.28	90.52	89.98
		5	90.10	89.86	90.10	89.94	**90.19**	89.99	89.91
SVM	Accuracy	300	91.78	92.14	91.59	91.19	87.45	91.29	**92.45**
		200	91.84	92.03	91.51	91.12	87.85	91.23	**92.50**
		100	91.71	92.03	91.93	90.78	78.87	91.20	**92.42**
		50	91.88	91.82	91.62	88.11	82.70	91.17	**92.24**
		30	91.68	91.58	91.87	88.11	90.29	91.32	**91.88**
		10	91.12	91.03	91.20	88.10	89.84	91.20	**91.29**
		5	90.87	90.80	90.84	88.48	90.90	**90.92**	90.80
LR	Accuracy	300	**92.83**	92.11	92.74	91.19	91.10	91.04	92.67
		200	92.60	92.04	92.56	91.11	91.07	91.04	**92.63**
		100	**92.54**	92.03	92.50	90.78	91.07	91.06	92.46
		50	**92.35**	91.79	92.32	88.07	91.04	90.92	92.29
		30	**92.03**	91.48	92.05	88.07	91.04	91.08	91.91
		10	**91.39**	91.04	91.40	88.08	91.08	90.85	91.30
		5	**90.90**	90.79	90.89	88.55	90.93	90.49	90.84

Table 7. Accuracy on IMDb dataset with different dimensionality reduction techniques and with fine-tuning. Best results are highlighted in bold.

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
K-NN	Accuracy	300	91.54	73.53	**91.56**	89.89	90.96	91.11	88.94
		200	91.62	77.74	**91.63**	90.43	90.80	91.28	89.62
		100	**91.56**	81.97	**91.56**	90.76	90.86	91.12	90.38
		50	91.66	83.93	**91.71**	90.59	90.78	91.09	90.91
		30	91.57	85.21	**91.72**	90.58	90.82	91.02	90.51
		10	91.23	89.36	**91.24**	90.66	90.99	90.91	90.57
		5	90.54	90.44	90.55	90.43	90.77	90.62	**90.86**
SVM	Accuracy	300	92.49	92.33	92.49	91.72	90.69	91.79	**92.56**
		200	92.07	92.26	92.30	91.67	87.30	91.61	**92.76**
		100	91.82	92.28	91.77	91.46	90.69	91.66	**92.82**
		50	92.21	92.11	91.79	90.58	88.09	91.73	**92.56**
		30	92.22	91.88	92.21	90.59	88.23	91.62	**92.29**
		10	91.74	91.64	91.63	90.58	91.43	91.37	**91.84**
		5	91.55	91.59	91.59	90.51	91.52	91.23	**91.53**
LR	Accuracy	300	92.86	92.32	92.90	91.72	91.69	91.52	**92.92**
		200	92.80	92.26	92.92	91.67	91.67	91.34	**92.99**
		100	**92.88**	92.26	92.84	91.46	91.64	91.34	92.82
		50	**92.71**	92.07	92.61	90.53	91.68	91.47	92.56
		30	92.46	91.87	**92.47**	90.57	91.66	91.23	92.33
		10	**93.10**	92.95	**93.10**	92.41	92.50	92.42	93.02
		5	**92.94**	92.87	**92.94**	92.42	92.48	92.57	92.82

5.2. BullyingV3.0 Dataset Results

In this section, we present results on BullyingV3.0, a dataset with a highly imbalanced class distribution that introduces new challenges for classification methods. To account for this imbalance, all tables report F1 score along with accuracy, which can be misleading in this context. The F1 score is a well-recognised metric, particularly valuable in imbalanced scenarios, as it balances precision and recall. While we have considered additional metrics such as ROC-AUC, we chose not to include them in this study. These metrics are more appropriate for in-depth class-level analysis, especially when the focus is on distinguishing false positives and false negatives. Our goal, however, is to use a consistent and established metric to evaluate how dimensionality reduction techniques perform across different algorithms and feature spaces. Including multiple metrics could introduce unnecessary complexity and noise, potentially distracting from the core contribution of our work.

Our first finding (Table 8) is that, without any dimensionality reduction, performance is substantially lower both with and without fine-tuning. This drop highlights the dataset classification complexity, even though its texts are comparatively shorter.

The trends observed in the IMDb dataset are also evident in these experiments. Our AE and VAE models consistently achieved the highest accuracy for dimensionality reduction, outperforming all other methods 100% of the time based on the F1 metric without any fine-tuning (Table 9), and maintaining this perfect performance after fine-tuning as well (Table 10). A key finding from our experimentation is that, consistent with observations on the IMDb dataset, dimensionality reduction can enhance classification performance even in the presence of class imbalance. Specifically, in the K-NN model without fine-tuning, the F1 score reaches 83.68%, which is equivalent to the performance of fine-tuned Logistic Regression combined with AE-based dimensionality reduction at 50 dimensions. This represents an approximate 10-point increase in F1 score compared to baseline models without feature reduction.

Furthermore, fine-tuning in conjunction with dimensionality reduction generally leads to performance improvements for algorithms such as Logistic Regression and SVM. Contrarily, for K-NN, fine-tuning appears to degrade classification performance relative to its non-fine-tuned counterpart.

Table 8. Performance metrics on the BullyingV3.0 dataset with and without fine-tuning. Best results are highlighted in bold.

Model	Without Fine-Tuning		With Fine-Tuning	
	Accuracy	F1	Accuracy	F1
K-NN	**84.49**	**70.66**	**87.75**	**75.64**
SVM	80.69	62.44	81.23	63.52
LR	82.69	64.93	82.86	65.64

5.3. Time and Performance Analysis

Although our main objective is to demonstrate that reducing the dimensionality of feature embeddings can maintain or even improve classification accuracy, we also aim to highlight another important benefit. By applying these techniques, we achieve shorter execution times. This not only enhances the applicability of real-time analysis and classification in social networks, where vast amounts of data are processed every second, but also ensures fast execution is essential. Additionally, it helps reduce the carbon footprint of our AI systems, contributing to more environmentally sustainable practices. For clarity, we report classification times only for the fine-tuned models, which achieved the best results.

We report execution time and CO_2 emissions for each algorithm combined with various dimensionality reduction techniques. Tables 11 and 12 present the results for the Bullying v3 dataset in terms of execution time and emissions, respectively. For the IMDb dataset, execution times are shown in Table 13, and emissions are reported in Table 14. In order to contrast if the differences between VAE and PCA are statistically relevant, we applied Wilcoxon signed-rank tests across all configurations. From these results, we found significant differences for the SVM classifier on the BullyingV3.0 dataset, with and without fine-tuning. Without fine-tuning, VAE significantly outperformed PCA ($p = 0.0138$), consistently obtaining higher accuracy across most dimensionality levels. When fine-tuning was applied, the improvement was more pronounced ($p = 0.0078$), reinforcing the robustness of VAE across different settings. These improvements are achieved alongside dimensionality reduction, thus suggesting that VAE can contribute to computational efficiency without compromising accuracy. These results, together with the results achieved with IMDb, confirm that VAE offers a statistically significant advantage over PCA in specific settings (in our case, specifically with SVM under fine-tuning), while remaining competitive in the remaining configurations, even under strong dimensionality constraints.

Table 9. Accuracyand F1 score on BullyingV3.0 dataset with different dimensionality reduction techniques and without fine-tuning. Best results are highlighted in bold.

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
K-NN	Accuracy	300	85.09	74.12	85.18	74.12	**87.06**	86.98	84.66
		200	85.78	74.12	85.86	74.12	**87.06**	86.72	85.43
		100	86.12	74.12	85.35	74.12	**87.32**	87.15	86.12
		50	86.46	74.12	86.72	74.12	87.15	**87.32**	86.03
		30	87.40	74.12	87.40	74.12	87.06	87.32	**87.57**
		10	86.89	74.12	86.89	74.12	**87.40**	86.89	86.46
		5	**86.80**	74.12	**86.80**	74.12	86.46	86.46	**86.80**
	F1	300	70.31	0.00	70.53	0.00	74.54	**83.06**	79.90
		200	71.48	0.00	71.70	0.00	74.71	**82.89**	80.81
		100	72.82	0.00	71.36	0.00	75.17	**83.35**	81.79
		50	73.49	0.00	73.95	0.00	74.75	**83.68**	81.53
		30	75.46	0.00	75.46	0.00	74.45	83.54	**83.75**
		10	74.02	0.00	74.02	0.00	75.38	**83.08**	82.24
		5	74.07	0.00	74.07	0.00	73.04	82.54	**82.61**
SVM	Accuracy	300	83.00	83.29	83.12	83.38	76.58	**83.86**	82.03
		200	83.32	82.95	81.81	83.38	80.35	**85.23**	82.23
		100	85.18	82.95	83.80	81.66	77.98	**85.43**	83.98
		50	83.55	82.35	82.92	80.72	68.24	**85.18**	84.26
		30	84.23	82.35	84.43	79.86	82.46	83.95	**84.83**
		10	84.92	81.92	84.92	74.12	82.23	84.92	**84.98**
		5	84.12	80.55	84.38	74.12	80.38	83.52	**85.72**
	F1	300	65.78	61.69	65.95	62.40	24.64	**78.60**	75.43
		200	67.88	60.44	63.16	62.11	47.75	**80.62**	76.02
		100	71.61	58.80	67.83	56.85	38.71	**80.69**	78.49
		50	65.74	56.72	66.52	54.55	40.54	**80.36**	79.09
		30	67.20	56.36	70.29	49.68	66.98	78.69	**80.06**
		10	71.01	54.62	69.43	0.00	68.95	**80.54**	80.49
		5	68.44	48.76	69.04	0.00	61.07	77.91	**81.49**

Table 9. *Cont.*

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
LR	Accuracy	300	83.29	74.12	82.78	74.12	**84.23**	83.98	83.38
		200	**84.66**	74.12	83.98	74.12	84.23	83.80	84.23
		100	**85.69**	74.12	84.06	74.12	84.23	84.40	84.75
		50	**85.43**	74.12	85.18	74.12	84.66	83.89	84.75
		30	84.75	74.12	84.83	74.12	84.15	83.12	**84.92**
		10	84.75	74.12	84.75	74.12	81.92	84.23	**85.52**
		5	84.15	74.12	84.15	74.12	83.12	82.69	**85.43**
	F1	300	66.44	0.00	65.04	0.00	70.98	**78.32**	77.95
		200	69.40	0.00	68.03	0.00	70.89	78.09	**78.99**
		100	71.84	0.00	67.82	0.00	71.16	79.03	**79.77**
		50	71.28	0.00	70.92	0.00	71.99	78.18	**79.58**
		30	69.93	0.00	69.95	0.00	71.05	77.00	**80.04**
		10	69.83	0.00	69.83	0.00	65.80	79.45	**80.98**
		5	67.71	0.00	67.71	0.00	67.55	76.61	**80.93**

Table 10. Accuracy and F1 score on BullyingV3.0 dataset with different dimensionality reduction techniques and with fine-tuning. Best results are highlighted in bold.

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
K-NN	Accuracy	300	83.29	74.12	82.78	74.12	**84.23**	83.98	83.38
		200	**84.66**	74.12	83.98	74.12	84.23	83.80	84.23
		100	**85.69**	74.12	84.06	74.12	84.23	84.40	84.75
		50	**85.43**	74.12	85.18	74.12	84.66	83.89	84.75
		30	84.75	74.12	84.83	74.12	84.15	83.12	**84.92**
		10	84.75	74.12	84.75	74.12	81.92	84.23	**85.52**
		5	84.15	74.12	84.15	74.12	83.12	82.69	**85.43**
	F1	300	66.44	0.00	65.04	0.00	70.98	**78.32**	77.95
		200	69.40	0.00	68.03	0.00	70.89	78.09	**78.99**
		100	71.84	0.00	67.82	0.00	71.16	79.03	**79.77**
		50	71.28	0.00	70.92	0.00	71.99	78.18	**79.58**
		30	69.93	0.00	69.95	0.00	71.05	77.00	**80.04**
		10	69.83	0.00	69.83	0.00	65.80	79.45	**80.98**
		5	67.71	0.00	67.71	0.00	67.55	76.61	**80.93**
SVM	Accuracy	300	83.32	85.00	83.38	86.03	86.80	**87.55**	83.29
		200	84.89	84.49	84.40	85.60	84.23	**87.09**	84.80
		100	84.60	85.69	84.86	85.09	70.95	**87.20**	85.63
		50	85.00	86.29	85.43	83.03	83.83	**87.12**	85.98
		30	85.18	86.20	85.32	83.55	83.15	**87.15**	86.52
		10	85.12	85.18	86.35	73.95	85.15	86.40	**86.80**
		5	87.03	85.43	85.12	74.12	85.75	86.20	**87.23**
	F1	300	64.88	66.02	66.30	69.42	73.95	**83.76**	77.78
		200	68.80	65.52	68.78	68.66	63.82	**83.23**	79.94
		100	69.17	68.55	69.86	67.29	42.25	**83.35**	81.10
		50	69.83	69.70	71.72	61.78	61.44	**83.23**	81.78
		30	68.10	69.57	69.49	63.36	59.96	**83.44**	82.30
		10	67.46	66.67	73.43	3.18	68.48	82.49	**82.88**
		5	74.80	67.18	67.61	0.00	72.76	82.33	**83.54**

Table 10. *Cont.*

Classifier	Metric	Dims	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
LR	Accuracy	300	85.09	74.12	85.18	74.12	**87.06**	86.98	84.66
		200	85.78	74.12	85.86	74.12	**87.06**	86.72	85.43
		100	86.12	74.12	85.35	74.12	**87.32**	87.15	86.12
		50	86.46	74.12	86.72	74.12	87.15	**87.32**	86.03
		30	87.40	74.12	87.40	74.12	87.06	87.32	**87.57**
		10	86.89	74.12	86.89	74.12	**87.40**	86.89	86.46
		5	**86.80**	74.12	**86.80**	74.12	86.46	86.46	**86.80**
	F1	300	70.31	0.00	70.53	0.00	74.54	**83.06**	79.90
		200	71.48	0.00	71.70	0.00	74.71	**82.89**	80.81
		100	72.82	0.00	71.36	0.00	75.17	**83.35**	81.79
		50	73.49	0.00	73.95	0.00	74.75	**83.68**	81.53
		30	75.46	0.00	75.46	0.00	74.45	83.54	**83.75**
		10	74.02	0.00	74.02	0.00	75.38	**83.08**	82.24
		5	74.07	0.00	74.07	0.00	73.04	82.54	**82.61**

It is important to note that the reported times refer to inference time over the entire dataset, and the gCO_2eq values represent the estimated carbon emissions specifically for that inference phase. Each value shown is the average of three independent runs. We did not include model training time in our analysis, as training is performed only once, while inference can be executed repeatedly. Our goal is to highlight that, once trained, the proposed system significantly improves inference efficiency, both in terms of time and environmental impact. It is also worth noting that in the classification time and emissions tables (Tables 11–14), the first three rows for each classification algorithm represent the inference time and carbon emissions measured without dimensionality reduction.

Table 11. Classification times for BullyingV3.0 dataset over different dimensionality reduction configurations. Best results are highlighted in bold.

Dims	Algorithm	No Red.	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
All dims	KNN	0.1122	-	-	-	-	-	-	-
	SVM	6.5781	-	-	-	-	-	-	-
	LR	0.2877	-	-	-	-	-	-	-
300	KNN	-	0.2375	5.1208	0.2660	2.3393	14.5920	0.0550	**0.0375**
	SVM	-	**0.3505**	5.2034	0.4411	2.3741	14.7715	1.8930	0.6578
	LR	-	0.2676	5.1130	0.3272	2.3298	14.6967	**0.1406**	0.1491
200	KNN	-	0.1881	2.6335	**0.1858**	2.1614	12.8356	0.0546	**0.0405**
	SVM	-	**0.2694**	2.6579	0.2864	2.1789	13.0293	2.1584	0.2703
	LR	-	0.2116	2.6270	0.2206	2.1571	12.9681	0.1590	**0.0923**
100	KNN	-	0.1414	1.0178	0.1333	1.8668	9.5506	0.0559	**0.0299**
	SVM	-	0.1672	1.0316	0.1594	1.8751	9.6212	1.5547	**0.0858**
	LR	-	0.1420	1.0239	0.1405	1.8616	9.5952	0.1611	**0.0558**
50	KNN	-	0.0962	0.4269	0.0633	1.6540	10.2038	0.0566	**0.0197**
	SVM	-	0.1012	0.4283	0.0733	1.6520	10.2410	1.9537	**0.0278**
	LR	-	0.0939	0.4234	0.0645	1.6504	10.2667	0.1876	**0.0170**
30	KNN	-	0.1045	0.3777	0.0558	2.0877	10.2379	0.0586	0.0451
	SVM	-	0.1012	0.3737	0.0560	2.0834	10.2539	1.5280	**0.0184**
	LR	-	0.1132	0.3728	0.0578	2.0828	10.2533	0.1632	**0.0178**
10	KNN	-	0.1315	0.3699	0.0880	2.6387	7.8423	**0.0678**	0.0980
	SVM	-	0.0542	0.2864	0.0343	2.5491	7.7853	1.5084	**0.0074**
	LR	-	0.0582	0.2881	0.0363	2.5526	7.8052	0.2418	**0.0170**
5	KNN	-	0.1498	0.3740	0.0993	1.2698	19.1309	0.0653	**0.0573**
	SVM	-	0.0660	0.3041	0.0357	1.2204	19.0851	1.2834	**0.0064**
	LR	-	0.7534	0.3064	0.0400	1.2235	19.0914	0.1730	**0.0219**

It is important to note that execution time is measured in seconds, while CO_2 emissions are reported in grams of CO_2 equivalent (gCO_2eq). To estimate carbon emissions, we used the CodeCarbon 3.0.4. a Python 3.7 library [68], which provides an automated and standardised method for assessing emissions from machine learning experiments. The library calculates emissions based on hardware configuration (CPU/GPU), runtime, power consumption, and geographical location, the latter to reflect the carbon intensity of the local energy grid.

The total CO_2 equivalent is computed according to Equation (2), where E_i represents the emissions of gas i (in grams or kilograms), and GWP_i denotes the Global Warming Potential of gas i over a 100-year time horizon. The result is expressed in terms of carbon dioxide equivalent (CO_2e):

$$CO_2e = \sum_i E_i \times GWP_i \tag{2}$$

This methodology enables a consistent and transparent evaluation of the environmental impact of each model training session. While absolute values may vary depending on infrastructure and regional energy profiles, CodeCarbon 3.0.4. offers a reliable framework for comparative analysis. To ensure fairness and reproducibility, all experiments were executed on the same hardware, and runtime was consistently recorded for each classification task.

Table 12. Evaluation of gCO_2eq emissions for BullyingV3.0 dataset over different dimensionality reduction configurations. Best entries are highlighted in bold.

Dims	Algorithm	No red.	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
All dims	KNN	0.000131	-	-	-	-	-	-	
	SVM	0.007633	-	-	-	-	-	-	
	LR	0.000335	-	-	-	-	-	-	
300	KNN	-	0.000918	0.012234	0.001150	0.005690	0.034384	0.000065	0.000044
	SVM	-	0.001122	0.012425	0.001435	0.005799	0.034654	0.002197	**0.000510**
	LR	-	0.001225	0.012513	0.001594	0.005864	0.034833	**0.000164**	0.000174
200	KNN	-	0.000764	0.006354	0.000875	0.005244	0.030273	0.000064	**0.000048**
	SVM	-	0.000924	0.006448	0.001081	0.005329	0.030552	0.002505	**0.000315**
	LR	-	0.001015	0.006505	0.001209	0.005392	0.030763	0.000185	**0.000108**
100	KNN	-	0.000556	0.002664	0.000547	0.004522	0.022547	0.000066	**0.000035**
	SVM	-	0.000653	0.002767	0.000641	0.004587	0.022686	0.001805	**0.000100**
	LR	-	0.000718	0.002849	0.000711	0.004643	0.022805	0.000188	**0.000066**
50	KNN	-	0.000410	0.001180	0.000337	0.004026	0.024046	0.000066	**0.000024**
	SVM	-	0.000481	0.001239	0.000400	0.004075	0.024151	0.002268	**0.000033**
	LR	-	0.000542	0.001292	0.000451	0.004126	0.024291	0.001970	**0.000020**
30	KNN	-	0.000412	0.001045	0.000358	0.005058	0.023976	0.000069	**0.000013**
	SVM	-	0.000465	0.001095	0.000428	0.005118	0.024050	0.001774	**0.000016**
	LR	-	0.000526	0.001140	0.000497	0.005180	0.024121	0.000190	**0.000022**
10	KNN	-	0.000467	0.001001	0.000364	0.006436	0.018336	**0.000079**	0.000115
	SVM	-	0.000505	0.001035	0.000403	0.006504	0.018370	0.001751	**0.000009**
	LR	-	0.000551	0.001071	0.000442	0.006595	0.018427	0.000281	**0.000016**
5	KNN	-	0.000579	0.001023	99.345446	0.003158	0.044515	0.000077	**0.000067**
	SVM	-	0.000622	0.001072	35.697301	0.003205	0.044549	0.001490	**0.000004**
	LR	-	0.000668	0.001115	40.028334	0.003255	0.044590	0.000202	**0.000010**

When analysing the results, it becomes evident that both processing time and CO_2 emissions are significantly reduced in the majority of cases. Exceptions to this trend are observed with a few models—namely, ICA, LLE, and UMAP—which introduce considerable overhead in terms of time and energy consumption. These exceptions will be analysed in more detail in Section 6.

Another important aspect to consider is the impact of the dataset specificities, particularly dataset size and average text length. In the case of the BullyingV3.0 dataset, which contains only 7321 examples with relatively short text sequences, the observed benefits of

dimensionality reduction are less pronounced. In contrast, for the IMDb dataset, which includes 50,000 samples with an average text length of 1289 characters, both time and CO_2 consumption are markedly higher. This highlights that the efficiency gains provided by dimensionality reduction techniques are more apparent—and more necessary—for large-scale datasets.

Table 13. Classification times for IMDb dataset over different dimensionality reduction configurations. Best entries are highlighted in bold.

Dims	Algorithm	No Reduction	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
All dims	KNN	8.1208	-	-	-	-	-	-	-
	SVM	6.6779	-	-	-	-	-	-	-
	LR	1.7711	-	-	-	-	-	-	-
300	KNN	-	3.3185	18.0089	3.9497	640.9010	58.9425	5.8711	**2.8873**
	SVM	-	**1.1295**	15.5802	2.0608	638.3636	57.7811	3.2424	4.9062
	LR	-	0.6804	15.2336	1.6244	638.1153	57.0314	0.8763	**0.5770**
200	KNN	-	2.5147	10.6664	2.6874	723.6974	45.4347	5.8749	**1.8997**
	SVM	-	**0.7724**	8.9030	1.3882	721.6401	44.7845	3.8942	1.6256
	LR	-	0.4714	8.6735	1.0941	721.4694	43.8987	0.7567	**0.3423**
100	KNN	-	1.7117	5.0284	2.0065	766.1196	30.7187	6.0608	1.4062
	SVM	-	0.5423	4.0499	0.8980	765.1256	30.1565	3.5009	**0.4789**
	LR	-	0.3981	3.9087	0.7337	765.0445	29.7182	0.9380	**0.1253**
50	KNN	-	1.3405	3.7035	1.1829	706.0574	25.0885	6.2141	**0.8375**
	SVM	-	0.4213	2.9186	0.5163	705.2746	24.6350	4.3489	**0.1271**
	LR	-	0.3264	2.8494	0.4419	705.1999	24.8079	0.9799	**0.0542**
30	KNN	-	1.3077	4.9349	1.0649	715.8028	23.1674	6.7874	**0.7683**
	SVM	-	0.3767	4.0024	0.4025	715.1210	22.6796	4.7007	**0.0752**
	LR	-	0.3046	3.9717	0.3659	715.0768	22.4977	1.0499	**0.0348**
10	KNN	-	3.9415	9.0696	**3.6465**	797.4667	19.4872	6.0613	5.3522
	SVM	-	0.1942	2.8076	0.2275	796.2334	18.2937	4.2129	**0.0284**
	LR	-	0.1916	2.8012	0.2246	796.2219	18.3165	0.9183	**0.0243**
5	KNN	-	1.2712	3.5514	1.4474	288.9107	46.4812	6.7752	**1.3948**
	SVM	-	0.2751	2.2860	0.2257	288.0114	45.0791	10.0579	**0.0206**
	LR	-	0.2811	2.2815	0.2245	288.0086	45.0876	1.6542	**0.0198**

In smaller datasets such as BullyingV3.0, the additional complexity introduced by incorporating a dimensionality reduction step may offset the time savings during classification. As a result, in some cases, the reduction in computational cost is not statistically significant. This highlights the trade-off between increased pipeline complexity and performance gains, which becomes particularly relevant in small or low-resource scenarios.

Using PCA with 100 dimensions, KNN classification time drops from 8.12 s to 1.71 s, achieving a 78.9% reduction. Similarly, with AE, times vary depending on the configuration but remain significantly below the baseline in most cases. However, VAE consistently achieves the most important improvements. For LR with 100 dimensions, classification time drops from 1.77 s to 0.125, a reduction of 92.9%. In extreme cases, such as LR with VAE at five dimensions, the time decreases to 0.0198s, representing a 98.9% reduction. As shown in Tables 13 and 14, dimensionality reduction techniques, especially PCA, AE, and VA, not only lead to significantly faster classification times but also to substantial reductions in carbon emissions during inference.

For example, in the KNN classifier, applying VAE at 50 dimensions reduces classification time from 8.12 s to 0.837 s (a ∼90% reduction) and cuts emissions from 0.0429 gCO_2eq to 0.0072 gCO_2eq, representing a ∼83% drop. Similarly, with Logistic Regression, using VAE at 10 dimensions brings time down from 1.77 s to 0.024 s (a ∼98.6% reduction) and emissions from 0.0438 gCO_2eq to 0.0011 gCO_2eq, saving nearly 97.5% in energy-related emissions. These trends are consistent across PCA and AE as well; however, VAE consistently offers the most significant efficiency gains, particularly at lower dimensionalities. For instance, PCA at 100 dimensions for LR reduces classification time by ∼77% and emissions by ∼88%.

Table 14. Evaluation of gCO_2eq emissions for IMDb dataset over different dimensionality reduction configurations. Best entries are highlighted in bold.

Dims	Algorithm	No Reduction	PCA	ICA	TSVD	LLE	UMAP	AE	VAE
All dims	KNN	0.042906	-	-	-	-	-	-	-
	SVM	0.043543	-	-	-	-	-	-	-
	LR	0.043796	-	-	-	-	-	-	-
300	KNN	-	0.009368	0.042906	0.010773	1.487585	1.487585	0.140200	**0.002241**
	SVM	-	0.010381	0.043543	0.011774	1.488009	1.488009	0.142522	**0.002522**
	LR	-	0.010908	0.043796	0.012279	1.488143	1.488143	0.143989	**0.000350**
200	KNN	-	0.006926	0.025502	0.007325	1.679929	1.679929	0.107997	**0.004590**
	SVM	-	0.007585	0.025972	0.007973	1.680336	1.680336	0.109658	**0.003002**
	LR	-	0.007901	0.026177	0.008291	1.680484	1.680484	0.110326	**0.000282**
100	KNN	-	**0.004711**	0.012305	0.005255	1.778402	1.778402	0.073428	0.007032
	SVM	-	0.005074	0.012605	0.005653	1.778592	1.778592	0.074597	**0.004063**
	LR	-	0.005261	0.012735	0.005857	1.778694	1.778694	0.075251	**0.001089**
50	KNN	-	**0.003778**	0.009112	0.003362	1.639152	1.639152	0.059961	0.007211
	SVM	-	**0.004040**	0.009313	0.003608	1.639340	1.639340	0.060681	0.005047
	LR	-	0.004188	0.009435	0.003763	1.639449	1.639449	0.061619	**0.001138**
30	KNN	-	0.003809	0.012082	**0.002975**	1.661700	1.661700	0.054487	0.007876
	SVM	-	0.004047	0.012243	**0.003162**	1.661868	1.661868	0.054872	0.005455
	LR	-	0.004203	0.012375	0.003301	1.661981	1.661981	0.055044	**0.001219**
10	KNN	-	0.009230	0.021362	0.008341	1.851171	1.851171	0.045616	**0.007034**
	SVM	-	0.009327	0.021463	0.008436	1.851276	1.851276	0.045723	**0.004889**
	LR	-	0.009421	0.021573	0.008530	1.851384	1.851384	0.045858	**0.001066**
5	KNN	-	**0.003302**	0.008536	0.003812	0.670890	0.670890	0.108215	0.007861
	SVM	-	**0.003380**	0.008628	0.003906	0.670975	0.670975	0.108334	0.011671
	LR	-	0.003463	0.008708	0.004001	0.671065	0.671065	0.108464	**0.001920**

Focusing on the best classification results, the best performance on the IMDb dataset was achieved using a fine-tuned LR model with a VAE at 200 dimensions, reaching an accuracy of 92.99%. This marks an improvement of nearly 0.5 percentage points over the baseline. For the Bullying v3 dataset, where class imbalance makes the F1 score a more suitable metric, the highest performance was obtained using a fine-tuned SVM with an Ar at 300 dimensions, achieving an F1 score of 83.76%.

In terms of execution time, the efficiency improvements are also substantial. On IMDb, inference time was reduced from 1.77 s (without dimensionality reduction) to 0.34 s using VAE, an approximate 80.8% reduction. Similarly, for the Bullying v3 dataset, inference time dropped from 6.57 s to 1.89 s using dimensionality reduction, yielding a reduction of around 71.2%. This trend is mirrored in the carbon emissions, demonstrating that dimensionality reduction techniques offer meaningful gains in both performance and sustainability.

6. Discussion

In this paper, we have presented a comprehensive analysis of how various dimensionality reduction techniques impact the text classification pipeline. Our results demonstrate that DL-based methods can significantly reduce computational time and CO_2 emissions, and in some scenarios, also improve classification performance. This highlights that, in the context of textual data, retaining all features and dimensions is not always the most effective approach. Based on our findings, several important discussions and future directions emerge that call for further exploration.

In the case of smaller datasets, such as BullyingV3.0, the reduction in processing time and CO_2 emissions achieved through dimensionality reduction techniques is minimal and, in some instances, even counterproductive. This limited impact can be attributed to various factors, including interference from background processes on the computing system and the initial computational overhead introduced by certain algorithms. Additionally, the relatively small data volume and shorter text length reduce the potential efficiency gains.

However, when these same techniques are applied to moderately larger datasets—such as IMDb, which, while not massive, already contains 50,000 examples—the benefits become more apparent. This trend suggests that the scalability of dimensionality reduction methods is a key advantage, with their positive impact on computational efficiency likely to become even more pronounced in large-scale or industrial applications involving truly massive datasets.

Regarding the dimensionality reduction techniques evaluated, UMAP and LLE consistently exhibited the highest computational cost in terms of processing time, paradoxically increasing even as the number of dimensions was reduced. This behaviour highlights the algorithms' inherent complexity, especially in scenarios involving smaller-dimensional embeddings. However, unlike other high-cost methods, UMAP sometimes yields notable improvements in classification performance, which, coupled with its popularity and ease of integration, still makes it a valuable option in many practical applications. LLE displayed a more problematic profile. Its computational time scaled poorly, likely due to the exponential growth in complexity associated with creating adjacency matrices. Moreover, in several experiments, LLE failed to deliver substantial classification gains, suggesting that its high cost is not always justified. These findings suggest that while some advanced reduction techniques may appear theoretically appealing, their practical applicability must be assessed in light of both computational overhead and predictive utility. In datasets with limited size or computational constraints, algorithms like LLE may introduce unnecessary complexity without corresponding performance improvements; therefore, they should be used with caution or potentially avoided.

An in-depth analysis of the lowest-performing models reveals that LLE and ICA consistently yield both poor classification results and higher computational cost. LLE is particularly sensitive to noise because it relies on preserving local neighbourhood structures, an assumption that breaks down with noisy and sparse data, such as text from social media used in our experiments. These texts often contain informal language, abbreviations, and inconsistent patterns that distort local relationships. Similarly, ICA assumes that the underlying features are statistically independent, which rarely holds true in textual data where words are often correlated or context-dependent. This mismatch between method assumptions and data characteristics significantly reduces the effectiveness of these techniques for text classification, helping explain their underperformance in our experiments.

In contrast to the lowest-performing models, we found that non-linear approaches such as VAE and AE consistently achieved the best results. This performance aligns well with the context-dependent nature of textual data. Unlike linear dimensionality reduction techniques, AEs and VAEs are capable of capturing complex, non-linear relationships within the data, an essential property for text, where semantic meaning often depends on subtle contextual cues. Additionally, by learning to reconstruct input data through compression, these models preserve general and task-relevant features while reducing the influence of dataset-specific noise. This results in robust embeddings that generalise effectively across datasets with varying properties, such as differing document lengths or class distributions. These qualities make deep, non-linear models particularly suitable for processing user-generated text from social networks.

Practical Implications and Limitations

The results obtained in this study highlight the practical relevance of using dimensionality reduction techniques in real-world applications, particularly in the context of sustainable social network analysis. Social platforms generate vast volumes of data every minute, requiring significant computational resources to process and analyse them. This computational load directly contributes to carbon emissions, especially when deep

learning-based systems are involved, as they often require frequent retraining and large-scale updates.

Our experiments show that dimensionality reduction methods can maintain predictive performance while significantly lowering the carbon footprint associated with data processing. This has important implications for the development of more environmentally responsible AI systems. By reducing the size of the input space, these methods enable faster training, less memory usage, and lower energy consumption, all without compromising the overall effectiveness of the models. However, the use of dimensionality reduction also introduces certain limitations. Depending on the technique, some interpretability may be lost, especially when projecting features into latent spaces that are not human-interpretable.

It is noteworthy to highlight that our approach is domain-independent, this meaning that the general approach could be applied to many classification scenarios of different nature, field, and final application. However, the effectiveness of the dimensionality reduction technique may vary across tasks and domains, and there is a risk of discarding features that are relevant in specific contexts. Therefore, careful evaluation and domain-specific tuning are necessary when applying these methods into other domains, specifically practical scenarios.

Regarding the deployment of our model in real-world or real-time settings, this is one of the key strengths of our work. Once the models are trained, they can be used in conjunction with our dimensionality reduction techniques with very low latency, achieving faster response times. In practical terms, the reduced input size allows faster inference, which is critical for real-time applications. This not only enhances performance but also improves interpretability and significantly reduces environmental impact. Based on these benefits, we conclude that our models and approach can be effectively integrated into real-world data analysis pipelines.

7. Conclusions

In this paper, we investigate the application of DL techniques to improve the processing and analysis of textual data from social media. Specifically, we assessed the effect of dimensionality reduction on text embeddings generated by pre-trained language models, considering four approaches: raw embeddings, fine-tuned embeddings, and both with and without dimensionality reduction.

Our results demonstrate that dimensionality reduction can reduce embedding size while preserving, or even improving, classification performance. Fine-tuning proved especially effective when enough data was available, offsetting potential performance losses introduced by reduction techniques.

On the balanced dataset, PCA, FastICA, TruncatedSVD, and VAE achieved the best performance. For the imbalanced dataset, AE and VAE delivered superior F1 scores, highlighting their strength in handling minority classes, despite lower accuracy. UMAP and LLE generally showed the weakest performance.

Regarding performance times, we shown that dimensionality reduction techniques can significantly improve both execution and classification speeds, while also being more environmentally friendly by reducing the carbon footprint.

In future work, it is necessary to test alternative fine-tuning strategies, expand to other social platforms, and apply the methodology to multilingual datasets to evaluate cross-lingual applications in social networks.

Author Contributions: Conceptualization, K.G.-B. and A.M.-G.; methodology, K.G.-B. and A.M.-G.; validation, M.J.M.-B. and K.G.-B.; investigation, J.A.D.-G.; resources, A.M.-G. and K.G.-B.; data curation, A.M.-G. and K.G.-B.; writing—original draft preparation, J.A.D.-G., K.G.-B., and A.M.-G.; writing—review and editing, J.A.D.-G., K.G.-B., A.M.-G., and M.J.M.-B.; supervision, M.J.M.-B. and

K.G.-B.; project administration, K.G.-B.; funding acquisition, M.J.M.-B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the following Grants: Grant PID2021-123960OB-I00, funded by MCIN/AEI/10.13039/501100011033 and by ERDF/EU, and Grant TED2021-129402B-C21, funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. It was also funded by "Consejería de Transformación Económica, Industria, Conocimiento y Universidades de la Junta de Andalucía" through a pre-doctoral fellowship program (Grant Ref. PREDOC_00298).

Data Availability Statement: The data used in this study are publicly available. Trained models and associated weights can be shared upon request.

Acknowledgments: This research was partially supported by the following Grants: Grant PID2021-123960OB-I00, funded by MCIN/AEI/10.13039/501100011033 and by ERDF/EU, and Grant TED2021-129402B-C21, funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. It was also funded by "Consejería de Transformación Económica, Industria, Conocimiento y Universidades de la Junta de Andalucía" through a pre-doctoral fellowship program (Grant Ref. PREDOC_00298). Finally, the research reported in this paper is also funded by the European Union (BAG-INTEL project, grant agreement no. 101121309).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

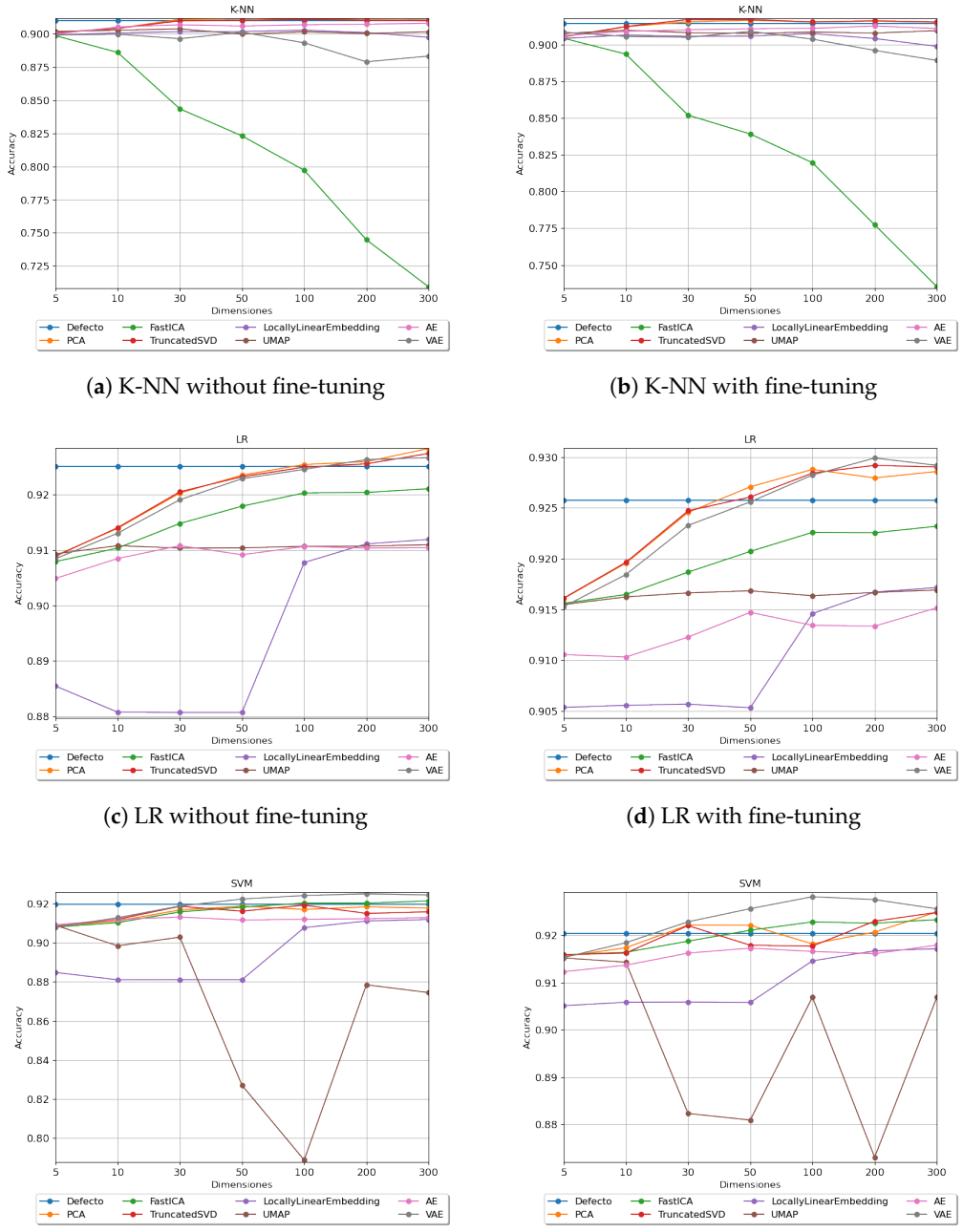
The following abbreviations are used in this manuscript:

AE	Autoencoder
Adam	Adaptive Moment Estimation
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
DCT	Discrete Cosine Transform
DL	Deep Learning
GloVe	Global Vectors for Word Representation
ICA	Independent Component Analysis
KL	Kullback–Leibler Divergence
LLE	Locally Linear Embedding
LLM	Large Language Model
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
KNN	K-Nearest Neighbour
NLP	Natural Language Processing
NMF	Non-negative Matrix Factorization
NPM	New Performance Metric
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
rRF	Removal of Redundant Feature
RNN	Recurrent Neural Network
SDG	Sustainable Development Goal
SOM	Self-Organising Map
SVD	Singular Value Decomposition
SVM	Support Vector Machine
t-SNE	t-distributed Stochastic Neighbour Embedding
UMAP	Uniform Manifold Approximation and Projection
VAE	Variational Autoencoder
Word2Vec	Word to Vector

Appendix A

Due to the extensive nature of our work and the variety of experiments conducted, we have included the graphical results of these experiments in this Appendix for reference, should they be needed. The lines in the graph represent different dimensionality reduction methods, either implemented or utilised, as described in the Proposed Approach section (Section 3).

Appendix A.1. IMDb Results



(**a**) K-NN without fine-tuning

(**b**) K-NN with fine-tuning

(**c**) LR without fine-tuning

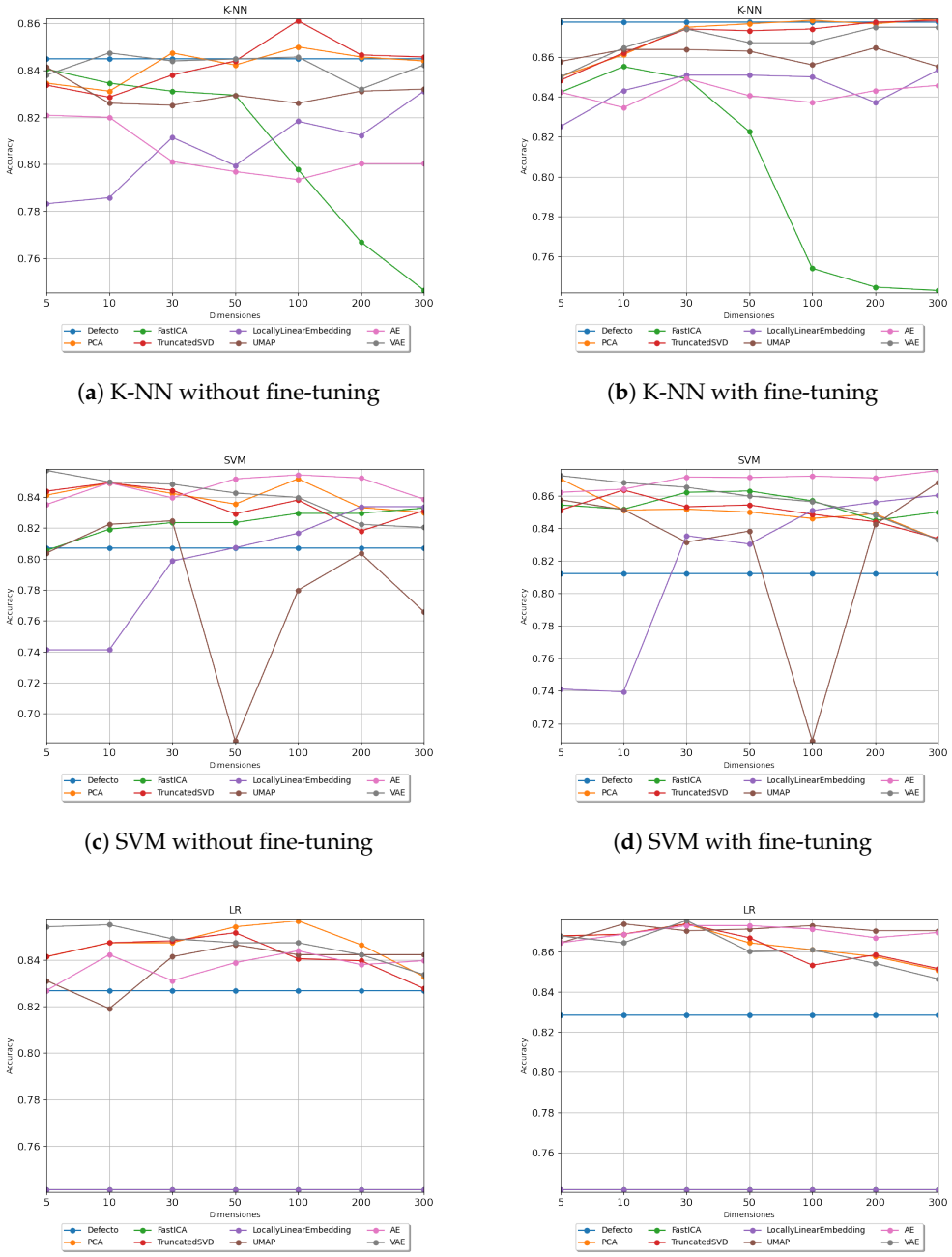
(**d**) LR with fine-tuning

(**e**) SVM without fine-tuning

(**f**) SVM with fine-tuning

Figure A1. IMDb dataset results for each classifier comparing without fine-tuning (**left**) and with fine-tuning (**right**).

Appendix A.2. BullyingV3.0 Results



(**a**) K-NN without fine-tuning (**b**) K-NN with fine-tuning

(**c**) SVM without fine-tuning (**d**) SVM with fine-tuning

(**e**) LR without fine-tuning (**f**) LR with fine-tuning

Figure A2. BullyingV3.0 dataset results for each classifier comparing without fine-tuning (**left**) and with fine-tuning (**right**).

References

1. Saad, A.I. Opinion mining on US Airline Twitter data using machine learning techniques. In Proceedings of the 2020 16th International Computer Engineering Conference (ICENCO), Cairo, Egypt, 29–30 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 59–63.
2. Das, R.; Singh, T.D. A step towards sentiment analysis of assamese news articles using lexical features. In Proceedings of the International Conference on Computing and Communication Systems: I3CS 2020, NEHU, Shillong, India, 28–30 April 2020; Springer: Singapore, 2021; pp. 15–23.
3. Rahat, A.M.; Kahir, A.; Masum, A.K.M. Comparison of Naive Bayes and SVM Algorithm based on sentiment analysis using review dataset. In Proceedings of the 2019 8th International Conference System Modeling and Advancement in Research Trends (SMART), Moradabad, India, 22–23 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 266–270.

4. Daud, S.; Ullah, M.; Rehman, A.; Saba, T.; Damaševičius, R.; Sattar, A. Topic classification of online news articles using optimized machine learning models. *Computers* **2023**, *12*, 16. [CrossRef]
5. Koufakou, A. Deep learning for opinion mining and topic classification of course reviews. *Educ. Inf. Technol.* **2024**, *29*, 2973–2997. [CrossRef]
6. Ghiassi, M.; Lee, S.; Gaikwad, S.R. Sentiment analysis and spam filtering using the YAC2 clustering algorithm with transferability. *Comput. Ind. Eng.* **2022**, *165*, 107959. [CrossRef]
7. Mageshkumar, N.; Vijayaraj, A.; Arunpriya, N.; Sangeetha, A. Efficient spam filtering through intelligent text modification detection using machine learning. *Mater. Today Proc.* **2022**, *64*, 848–858. [CrossRef]
8. Abid, M.A.; Ullah, S.; Siddique, M.A.; Mushtaq, M.F.; Aljedaani, W.; Rustam, F. Spam SMS filtering based on text features and supervised machine learning techniques. *Multimed. Tools Appl.* **2022**, *81*, 39853–39871. [CrossRef]
9. Supriyono; Wibawa, A.P.; Suyono; Kurniawan, F. Advancements in natural language processing: Implications, challenges, and future directions. *Telemat. Inform. Rep.* **2024**, *16*, 100173. [CrossRef]
10. Zareapoor, M.; Seeja, K.R. Feature Extraction or Feature Selection for Text Classification: A Case Study on Phishing Email Detection. *Int. J. Inf. Eng. Electron. Bus.* **2015**, *7*, 60–65. [CrossRef]
11. Kumar, K.V.; Srinivasan, R.; Singh, E.B. An efficient approach for dimensionality reduction and classification of high dimensional text documents. In Proceedings of the First International Conference on Data Science, E-Learning and Information Systems, Madrid, Spain, 1–2 October 2018; pp. 1–5.
12. Vieira, A.S.; Diz, M.L.B.; Iglesias, E.L. Improving the text classification using clustering and a novel HMM to reduce the dimensionality. *Comput. Methods Programs Biomed.* **2016**, *136*, 119–130. [CrossRef]
13. McAllister, R.; Sheppard, J. Taxonomic Dimensionality Reduction in Bayesian Text Classification. In Proceedings of the 2012 11th International Conference on Machine Learning and Applications, Boca Raton, FL, USA, 12–15 December 2012; IEEE: Piscataway, NJ, USA, 2012; Volume 1, pp. 508–513.
14. Akritidis, L.; Bozanis, P. How dimensionality reduction affects sentiment analysis NLP tasks: An experimental study. In Proceedings of the IFIP International Conference on Artificial Intelligence Applications and Innovations, Hersonissos, Crete, Greece, 17–20 June 2022; Springer: Cham, Switzerland, 2022; pp. 301–312.
15. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Chenaghlu, M.; Gao, J. Deep Learning–based Text Classification: A Comprehensive Review. *ACM Comput. Surv.* **2021**, *54*, 62. [CrossRef]
16. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 1–9.
17. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
18. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019.
19. Evangelista, P.F.; Embrechts, M.J.; Szymański, B.K. Taming the Curse of Dimensionality in Kernels and Novelty Detection. In Proceedings of the Online World Conference on Soft Computing in Industrial Applications, Online, 20 September–8 October 2004.
20. Hosseini, S.; Varzaneh, Z.A. Deep text clustering using stacked AutoEncoder. *Multimed. Tools Appl.* **2022**, *81*, 10861–10881. [CrossRef]
21. Daneshfar, F.; Soleymanbaigi, S.; Nafisi, A.; Yamini, P. Elastic deep autoencoder for text embedding clustering by an improved graph regularization. *Expert Syst. Appl.* **2024**, *238*, 121780. [CrossRef]
22. Heusinger, M.; Raab, C.; Schleif, F.M. Dimensionality reduction in the context of dynamic social media data streams. *Evol. Syst.* **2022**, *13*, 387–401. [CrossRef]
23. Khan, J.; Ahmad, K.; Jagatheesaperumal, S.K.; Sohn, K.A. Textual variations in social media text processing applications: Challenges, solutions, and trends. *Artif. Intell. Rev.* **2025**, *58*, 89. [CrossRef]
24. Singh, K.N.; Devi, S.D.; Devi, H.M.; Mahanta, A.K. A novel approach for dimension reduction using word embedding: An enhanced text classification approach. *Int. J. Inf. Manag. Data Insights* **2022**, *2*, 100061. [CrossRef]
25. Zheng, W.; Qian, Y. Aggressive dimensionality reduction with reinforcement local feature selection for text categorization. In Proceedings of the Artificial Intelligence and Computational Intelligence: International Conference, AICI 2010, Sanya, China, 23–24 October 2010; Proceedings, Part I 2; Springer: Berlin/Heidelberg, Germany, 2010; pp. 365–372.
26. Mohamed, A. An effective dimension reduction algorithm for clustering Arabic text. *Egypt. Inform. J.* **2020**, *21*, 1–5. [CrossRef]
27. Walkowiak, T.; Datko, S.; Maciejewski, H. Reduction of dimensionality of feature vectors in subject classification of text documents. In Proceedings of the Reliability and Statistics in Transportation and Communication: Selected Papers from the 18th International Conference on Reliability and Statistics in Transportation and Communication, RelStat'18, Riga, Latvia, 17–20 October 2018; Springer: Cham, Switzerland, 2019; pp. 159–167.

28. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
29. Elhadad, M.K.; Badran, K.M.; Salama, G.I.M. A novel approach for ontology-based dimensionality reduction for web text document classification. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; pp. 373–378.
30. Corrêa, R.F.; Ludermir, T.B. Dimensionality Reduction by Semantic Mapping in Text Categorization. In Proceedings of the International Conference on Neural Information Processing, Calcutta, India, 22–25 November 2004.
31. Swarnalatha, K.; Kumar, N.V.; Guru, D.S.; Anami, B.S. Analysis of Dimensionality Reduction Techniques for Effective Text Classification. In Proceedings of the 2021 International Conference on Intelligent Technologies (CONIT), Hubli, India, 25–27 June 2021; pp. 1–5.
32. Davy, M.; Luz, S. Dimensionality reduction for active learning with nearest neighbour classifier in text categorisation problems. In Proceedings of the Sixth International Conference on Machine Learning and Applications (ICMLA 2007), Cincinnati, OH, USA, 13–15 December 2007; pp. 292–297.
33. Chamorro-Padial, J.; Rodríguez-Sánchez, R. Text Categorisation Through Dimensionality Reduction Using Wavelet Transform. *J. Inf. Knowl. Manag.* **2020**, *19*, 2050039. [CrossRef]
34. Saarikoski, J.; Laurikkala, J.; Järvelin, K.; Siermala, M.; Juhola, M. Dimensionality reduction in text classification using scatter method. *Int. J. Data Min. Model. Manag.* **2014**, *6*, 1–21. [CrossRef]
35. Vieira, A.S.; Iglesias, E.L.; Diz, M.L.B. A New Dimensionality Reduction Technique Based on HMM for Boosting Document Classification. In Proceedings of the Practical Applications of Computational Biology & Bioinformatics, Salamanca, Spain, 3–5 June 2015.
36. Kim, H.; Howland, P.; Park, H. Dimension Reduction in Text Classification with Support Vector Machines. *J. Mach. Learn. Res.* **2005**, *6*, 37–53.
37. Yin, S.; Huang, Z.; Chen, L.; Qiu, Y. A Approach for Text Classification Feature Dimensionality Reduction and Rule Generation on Rough Set. In Proceedings of the 2008 3rd International Conference on Innovative Computing Information and Control, Dalian, China, 18–20 June 2008; p. 554.
38. Durmaz, O.; Bilge, H.S. Effects of dimensionality reduction and feature selection in text classification. In Proceedings of the 2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 20–22 April 2011; pp. 21–24.
39. Jain, G.; Ginwala, A.; Aslandogan, A. An approach to text classification using dimensionality reduction and combination of classifiers. In Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, 2004. IRI 2004, Las Vegas, NV, USA, 8–10 November 2004; pp. 564–569.
40. Dong, W. Mixed feature dimension reduction strategy for text categorization. *J. Guizhou Norm. Coll.* **2012**, *28*, 6–10.
41. Boyapati, M.; Aygun, R. Semanformer: Semantics-aware Embedding Dimensionality Reduction Using Transformer-Based Models. In Proceedings of the 2024 IEEE 18th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 5–7 February 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 134–141.
42. Thor, F.; Nettelblad, C. Dimensionality reduction of genetic data using contrastive learning. *Genetics* **2025**, iyaf068. [CrossRef] [PubMed]
43. Nareklishvili, M.; Geitle, M. Deep ensemble transformers for dimensionality reduction. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *36*, 2091–2102. [CrossRef] [PubMed]
44. Sakr, C.; Khailany, B. Espace: Dimensionality reduction of activations for model compression. *arXiv* **2024**, arXiv:2410.05437. [CrossRef]
45. Khan, A.; Majumdar, D.; Mondal, B. Sentiment analysis of emoji fused reviews using machine learning and Bert. *Sci. Rep.* **2025**, *15*, 7538. [CrossRef] [PubMed]
46. Dandannavar, P.; Mangalwede, S.; Deshpande, S. Emoticons and their effects on sentiment analysis of Twitter data. In Proceedings of the EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing: BDCC 2018, Coimbatore, India, 13–15 December 2018; Springer: Cham, Switzerland, 2020; pp. 191–201.
47. Maas, A.L.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 142–150.
48. Bullying v3 Dataset. n.d. University of Wisconsin–Madison. Available online: https://research.cs.wisc.edu/bullying/data.html (accessed on 23 July 2025).
49. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep Contextualized Word Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 2227–2237.

50. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018. Available online: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf (accessed on 23 July 2025).
51. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Hong Kong, China, 3–7 November 2019; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 1–11. [CrossRef]
52. Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; Krishnan, D. Supervised contrastive learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 18661–18673.
53. Jolliffe, I.T. Principal component analysis for special types of data. In *Principal Component Analysis*; Springer Series in Statistics; Springer: New York, NY, USA, 2002.
54. Comon, P. Independent component analysis, a new concept? *Signal Process.* **1994**, *36*, 287–314. [CrossRef]
55. Roweis, S.T.; Saul, L.K. Nonlinear dimensionality reduction by locally linear embedding. *Science* **2000**, *290*, 2323–2326. [CrossRef]
56. Hansen, P.C. The truncated SVD as a method for regularization. *BIT Numer. Math.* **1987**, *27*, 534–553. [CrossRef]
57. McInnes, L.; Healy, J.; Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv* **2018**, arXiv:1802.03426.
58. Cover, T.; Hart, P. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **1967**, *13*, 21–27. [CrossRef]
59. McCullagh, P.; Nelder, J.A. *Generalized Linear Models*; Springer: Boston, MA, USA, 1989. [CrossRef]
60. Chang, C.C.; Lin, C.J. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27. [CrossRef]
61. Wang, H.; Zhang, H.; Yu, D. On the dimensionality of sentence embeddings. *arXiv* **2023**, arXiv:2310.15285. [CrossRef]
62. sentence-transformers/all-mpnet-base-v2 · Hugging Face. Available online: https://huggingface.co/sentence-transformers/all-mpnet-base-v2 (accessed on 3 June 2025).
63. sentence-transformers/all-distilroberta-v1 · Hugging Face. Available online: https://huggingface.co/sentence-transformers/all-distilroberta-v1 (accessed on 3 June 2025).
64. Song, K.; Tan, X.; Qin, T.; Lu, J.; Liu, T.Y. Mpnet: Masked and permuted pre-training for language understanding. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 16857–16867.
65. Pretrained Models—Sentence-Transformers Documentation. Available online: https://www.sbert.net/docs/pretrained_models.html (accessed on 3 June 2025).
66. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
67. Wilcoxon, F. Individual comparisons by ranking methods. *Biom. Bull.* **1945**, *1*, 80–83. [CrossRef]
68. CodeCarbon: Track Carbon Emissions from Machine Learning Training. 2021. Available online: https://github.com/mlco2/codecarbon (accessed on 29 May 2025).