The Indonesian Journal of Computer Science
www.ijcs.net
Volume 14, Issue 2, April 2025
https://doi.org/10.33022/ijcs.v14i2.4740

Examining 1D and 2D CNN Architectures in Comparison for Sentiment Analysis in Sequential Data: A Case Study of Spotify Music Reviews

Courage Matobobo[1], Tinashe Crispen Garidzira[2]
cmatobobo@wsu.ac.za[1], crispengari@gmail.com[2]
[1]Walter Sisulu University
[2]Independent Researcher

Article Information

Received : 14 Feb 2025
Revised : 21 Mar 2025
Accepted : 15 Apr 2025

Keywords

Sequential Data Processing (SDP), Convolutional Neural Networks (CNNs), Sentiment Analysis, Spotify Music Reviews, Text Classification

Abstract

This study examines the comparative performance of one-dimensional (1D) and two-dimensional (2D) Convolutional Neural Networks (CNNs) in processing sequential data for sentiment analysis, using Spotify music reviews as a case study. Leveraging a custom dataset from Kaggle, the study examines the effectiveness of CNN architectures in extracting meaningful patterns from text input. The study integrates PyTorch and TorchText for efficient data preprocessing and model deployment. Both architectures are evaluated based on classification accuracy, computational efficiency, and ability to handle sequential dependencies. The results highlight the strengths and limitations of each method, providing insight into their suitability for similar tasks in text-based sentiment analysis. This research provides valuable guidance for researchers and practitioners working on sequential data tasks, emphasizing the role of architectural design in achieving optimal performance.

A. Introduction

Sentiment analysis is a critical aspect of natural language processing (NLP) to understand and interpret people's opinions, emotions, and attitudes about a certain product, or service, or entity. In recent years, sentiment analysis has been accepted by various businesses, governments, and organisations to analyse user-generated content and derive actionable insights [1], [2]. The exponential rise of user-generated content on platforms like Spotify has improved sentiment analysis's use to analyse sentiments in music reviews [3].

This study examines the application of one-dimensional (1D) and two-dimensional (2D) convolutional neural network (CNN) architectures for sentiment analysis in sequential data, focusing on Spotify music reviews as a case study. While CNNs are primarily employed for image data processing, their versatility in handling text data has enabled robust performance on various NLP tasks, including sentiment analysis. CNN applies convolutional filters that automatically learn features suitable for the given task, such as sentiment indicators, key phrases, and contextual features [4]. CNN may be implemented in either 1D or 2D, depending on how input data is processed [5]. For example, 1D is commonly used in NLP and text processing, while 2D is more common in image processing but can be applied to text in specific cases [6].

Despite the widespread use of CNNs in sentiment analysis, the comparative performance of 1D and 2D CNN architectures for sentiment analysis on sequential data, particularly in emotionally rich and context-dependent domains such as music reviews, remains underexplored. Music reviews present unique challenges due to their subjective and emotional nature, often characterised by abstract descriptors such as "soulful" or "melancholic," which differ from functional language found in product or move reviews [7], [8]. This study, therefore, addresses this gap by evaluating the performance of 1D and 2D CNNs on Spotify music reviews to identify the strengths and weaknesses of each architecture in capturing the nuanced sentiments in this domain.

Sentiment analysis (SA), also known as opinion mining, is a field of study that investigates people's sentiments on things, including events, subjects, people, problems, services, goods, organizations, and their characteristics [9], [10]. Sentiment analysis is a fundamental technique in NLP. It classifies text as positive, negative, or neutral. Sentiment analysis has been applied in various fields, including marketing, stock market prediction, politics, healthcare, and religious organisations [9], [11], [12], [13],[14]. Traditional approaches to sentiment analysis, such as rule-based and lexicon-based methods, have limitations in handling complex language constructs like sarcasm, irony, and context-dependent expressions [9], [15]. Machine learning approaches, such as Naïve Bayes and Support Vector Machines, have improved sentiment classification but often require large datasets and careful feature engineering [9].

The coming of deep learning has transformed sentiment analysis, making models automatically learn and extract complex data patterns. Deep learning architectures, like CNN, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs), have demonstrated superior performance in capturing context nuances and handling multilingual applications ([16], [17]. However, each architecture has some weaknesses. For instance, RNNs

and LSTMs struggle with long-range dependencies and noisy data, while GRUs face challenges in interpretability and computational efficiency [18], [19]. CNNs, on the other hand, offer computational efficiency and the ability to capture local features, making them particularly suitable for text-based sentiment analysis [20], [21].

In the music industry, sentiment analysis is crucial as it helps understand audience preferences, predict song popularity, and structure marketing strategies [22]. However, music reviews' emotionally rich and context-dependent nature poses unique challenges for sentiment analysis, necessitating domain-specif approach reviews [7], [8]. The study addresses several gaps in the literature: (i) limited comparisons of 1D and 2D CNN for sequential data, particularly in the context of sentiment analysis; (ii) the lack of research on Spotify reviews as a dataset, which presents a unique challenge because of emotionally rich and context-dependent language; and (iii) the need for a thorough performance evaluation framework to evaluate CNN models in sentiment analysis tasks in a systematic manner [9].

This study bridges these gaps by comparing 1D and 2D CNNs for sentiment analysis on Spotify music reviews, providing insights into their strengths and limitations. Addressing these gaps contributes to the broader understanding of CNN architectures for sequential data and offers practical implications for sentiment analysis in domain-specific applications.

The rest of the paper is structured as follows: Section B discusses the methodology adopted, Section C presents the results and discusses the findings, and Section D concludes the paper.

B. Research Method

The researchers used the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology to structure the data mining process in this study. This framework comprises six main phases: business understanding, data understanding, data preparation, modelling, evaluation, and deployment [23]. By adopting this methodology, the study ensured a systematic and organized approach to data mining, enabling effective project management and efficient resource allocation.

The researchers utilized a dataset titled “Spotify User Reviews,” sourced from Kaggle, a publicly accessible database for datasets [24]. The dataset comprises 51,473 music reviews, with 55.83% classified as negative and 44.17% as positive. It includes two columns: Review (containing the review content) and Label (indicating the sentiment label) [25]. Figure 1 shows the distribution of class labels before and after the dataset was balanced.

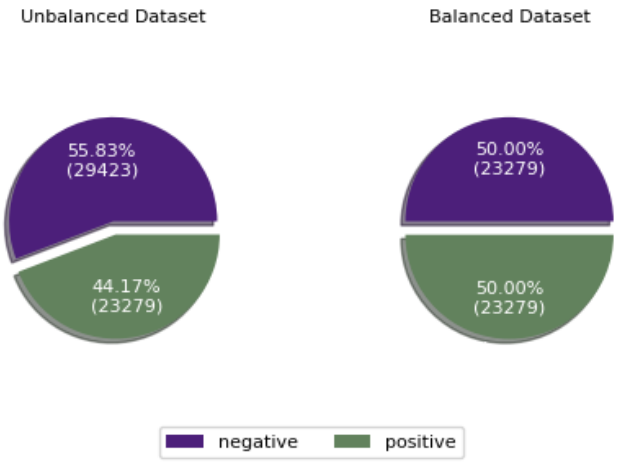

Figure 1. Data distribution in the original dataset

Figure 1 shows the data distribution in the original dataset and the adjusted distribution after balancing the labels, particularly addressing the underrepresented positive reviews. Post-balancing, each class comprises 23,279 paired examples of reviews with their respective sentiment labels. Balancing the dataset is crucial for classification tasks because an imbalanced dataset can lead to biased model performance and reduced model generalization, where the classifier becomes skewed towards the majority class, making classification accuracy an unreliable measure of model performance [26]. This bias can result in poor generalization to minority classes, thereby affecting the overall performance of the model [27]. Recent studies have emphasized the importance of addressing class imbalance to improve model robustness and ensure fair evaluation metrics [28].

After balancing the dataset based on the minority class, the dataset was split into four subsets: training, testing, validation, and inference. While it is common in machine learning to split datasets into three subsets (training, testing, and validation), the researchers introduced an additional inference set to enhance the model's practical applicability. The training set was used to train the model, the validation set was used to fine-tune hyperparameters and validate performance during training, and the testing set was used to evaluate the model's final performance after training. The inference set, a unique addition, was reserved for making predictions using the best-trained model, simulating real-world deployment scenarios [25], [29]. Figure 2 shows the distribution of examples across these subsets using pie charts.

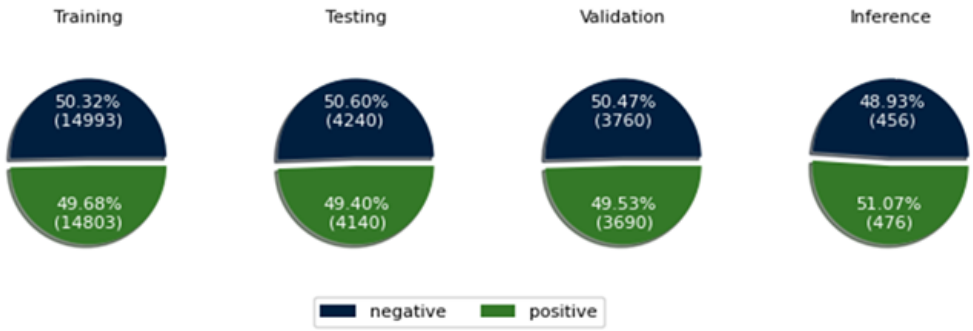

Figure 2. Distribution of labels in each of the 4 subsets

Figure 2 shows how the samples were distributed in each set after splitting the dataset into 4 subsets. After splitting, the training dataset consists of 14,993 negative and 14,803 positive reviews. The testing set contains 4,140 positive reviews and 4,240 negative reviews, while the validation set includes 3,760 positive and 3,690 negative reviews. Lastly, the inference set comprises 456 negative reviews and 476 positive reviews. Dataset splitting is a critical step in machine learning, ensuring the model can generalize to unseen data. Researchers can prevent overfitting by separating data into training, validation, and testing sets, where a model performs well on training data but poorly on new data. The training set allows the model to learn patterns, the validation set helps optimize hyperparameters, and the testing set provides an unbiased evaluation of the model's performance [29]. This structured approach ensures robustness and reliability in machine learning workflows. Figure 3 shows the fractions of each subset in the entire dataset.

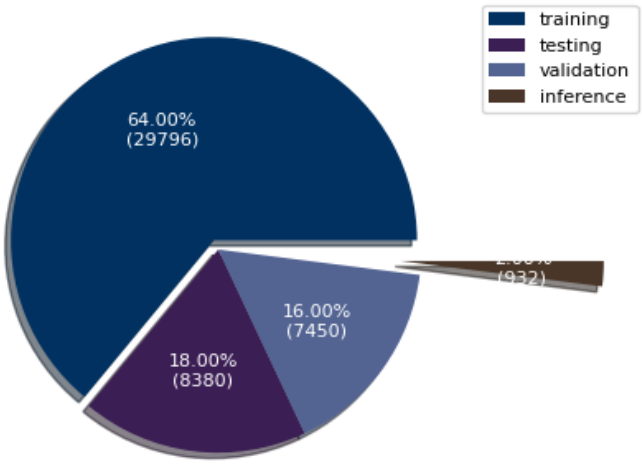

Figure 3. Fraction of each sunset from the entire dataset

Figure 3 illustrates the distribution of examples across each subset. Most reviews belong to the training set (64%), followed by the testing set, which comprises 18% of the dataset. The validation set accounts for 16%, while the remaining fraction was allocated to the inference set. Before cleaning and normalizing the text reviews, researchers analysed the most frequently occurring words in the music reviews. Figure 4 shows a visualization of the most common words appearing in each subsets using a word cloud plot.

Figure 4. Most common words before text cleaning

Figure 4 shows the most common words in the dataset before text cleaning. It was observed that the most frequently occurring words in the reviews were stopwords, such as “the”, “to” and “is”. Stopwords are common words in a language that carry little to no meaningful information for text analysis or modeling [30]. These stopwords were removed from each review as part of the text-cleaning process. Stopwords like “the”, “to”, and were among the most prevalent in the dataset but did not contribute significantly to the model's understanding of the text's sentiment or context. Removing stopwords is a crucial preprocessing step in NLP because it helps reduce noise in the data and improves computational efficiency. Stopwords do not provide meaningful insights for tasks like sentiment analysis or topic modelling, and their removal allows the model to focus on more informative words, enhancing its performance and accuracy [31]. This step also reduces the dimensionality of the dataset, making it easier for the model to process and learn from the data. Figure 5 shows the most common words that appeared in the reviews after the reviews were cleaned by removing stop words and some punctuation.

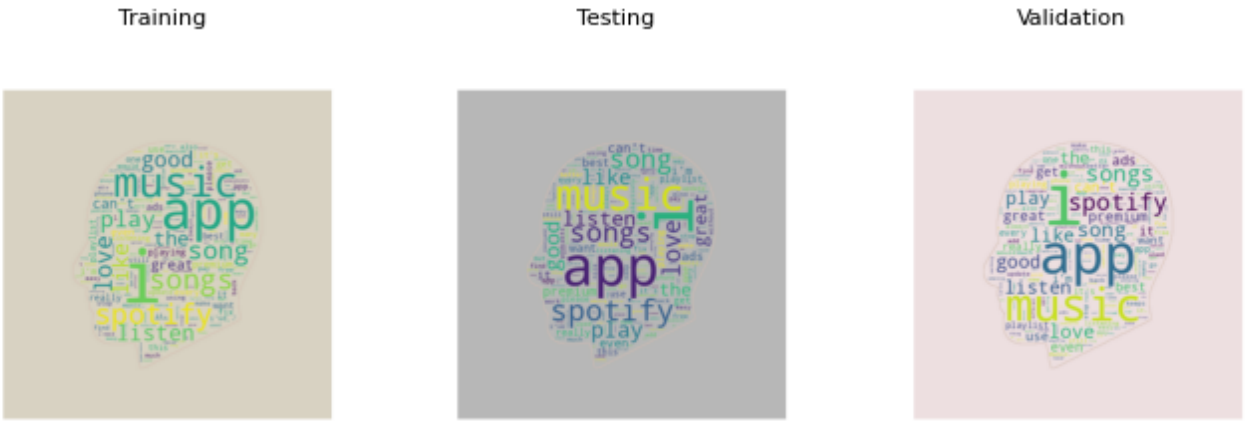

Figure 5. Most common words after text cleaning

Figure 5 shows the distribution of most common words in the training, testing and validation sets after stop words was removed and after the review text

were cleaned and normalized. It can be observed that the most frequently occurring words in the reviews include “music,” “app,” “songs,” and “Spotify,” among others. This supports the hypothesis that the reviews relate to the Spotify music app. After cleaning the reviews, the text was tokenized using the spaCy English tokenizer model to prepare for vocabulary creation. A vocabulary is a word-to-integer mapping that acts as a lookup dictionary, assigning each word a unique integer value. This step is critical because machine learning models cannot process raw text directly; they require numerical inputs [32]. During vocabulary creation, a minimum frequency threshold of 2 was set, meaning words appearing fewer than twice in the dataset were replaced with an unknown token (“unk”). Using the torchtext vocab function, a vocabulary was generated based on the training data, with four special tokens: “unk” (unknown), “pad” (padding), “sos” (start of sequence), and “eos” (end of sequence). The resulting vocabulary size was 7,514, representing the number of unique words in the corpus [31], [33].

The textual labels in the dataset were converted into numerical values by creating a labels dictionary, where 0 represented positive reviews, and 1 represented negative reviews. Pretrained word embeddings were utilized to enhance the model's performance. Specifically, the GloVe.6B.50d embeddings were downloaded—a 50-dimensional GloVe model trained on approximately 6 billion words [34]. An embedding matrix was then constructed to align the downloaded embeddings with the vocabulary, ensuring compatibility with the vocabulary size of 7,514. Finally, the Spotify Reviews dataset was organized into batches of 128 samples each, preparing it for input into the CNN model [29], [35].

The models were developed using the PyTorch library in Python, employing the same base architecture but varying the number of CNN layers to enable performance comparison. CNNs are widely used in text classification tasks because they capture data's local patterns and hierarchical features [31]. Figure 6 illustrates the general architecture of the CNN models created in this study.

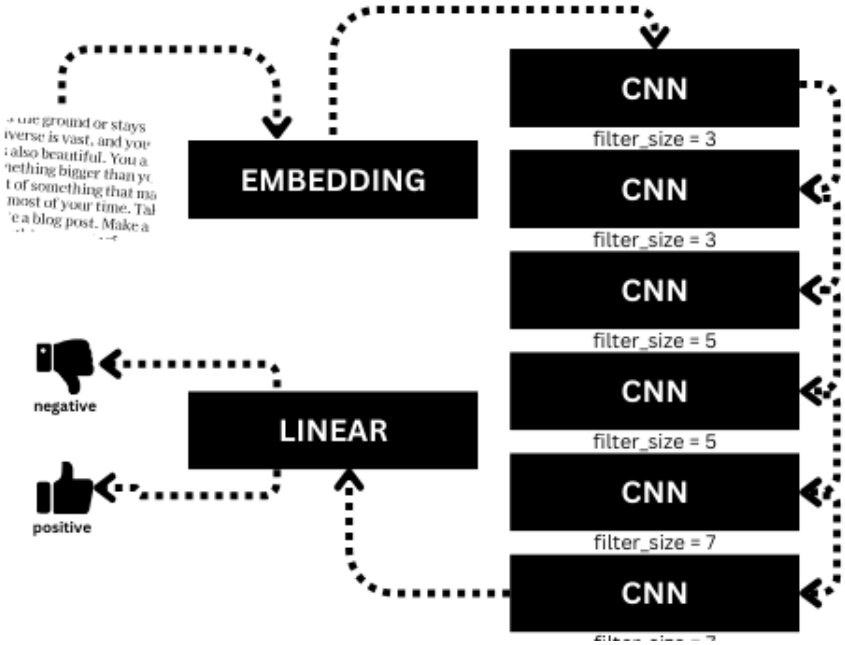

Figure 6. CNN Models' general architecture

Figure 6 illustrates the general architecture of the two models. The first layer in both models is an embedding layer with an embedding size of 50, corresponding to the dimensionality of the pre-trained GloVe word embeddings used in this study. This embedding layer is encapsulated within a Sequential layer. Following this, another Sequential layer, wrapped within a ModuleList, contains six CNN layers with filter sizes of [3, 3, 5, 7, 7, 7], respectively. These layers are stacked sequentially to extract hierarchical features from the input data. The final layer of the neural network is a fully connected layer with an output dimension of 1, suitable for the binary classification task at hand [31].

The output from the embedding layer is passed through the CNN layers, where a ReLU activation function is applied to introduce non-linearity. The outputs of the CNN layers are then pooled using the max_pool_1D function to reduce dimensionality and retain the most salient features. A dropout layer with a probability of 0.67789 is applied to the pooled output to mitigate overfitting before it is forwarded to the output layer during the forward pass [36]. Once the models were created and initialized, they were transferred to a GPU device on Google Colab for training. The total number of trainable parameters for each model architecture was calculated, and Table 1 summarizes these parameters for comparison.

Table 1. Models' Parameters

CNN MODEL	*TOTAL PARAMETERS*	*TRAINABLE PARAMETERS*
1D	451 301	451 301
2D	451 301	451 301

Table 1 indicates that the 1D and 2D Convolutional Neural Network (CNN) models possess an equal number of parameters, totalling 451,301, encompassing both trainable and non-trainable parameters. Specific initialization functions enhanced training efficiency instead of randomly initializing weights. For layers classified as Linear, weights were initialized using the Xavier Normal distribution, defined as:

$$w \sim N(0, \frac{1}{n_{in}}) \quad (1)$$

Here, n_in represents the number of input neurons to the layer. This method aims to maintain a consistent variance of activations across layers, thereby preventing issues related to vanishing or exploding gradients during training [37]. Biases in these layers were initialized to zero. For Convolutional layers, biases were similarly set to zero, while weights were initialized using the Kaiming Normal distribution:

$$w \sim N(0, \frac{2}{n_{in}}) \quad (2)$$

This approach is particularly effective for layers utilizing ReLU activation functions, as it accounts for the non-linearity introduced by ReLU, ensuring that the variance of activations remains stable throughout the network [38]. Pretrained embedding matrices were utilized to avoid random initialization for the embedding layers in all models, thereby leveraging prior knowledge and

potentially enhancing model performance [33]. Each model was optimized using the Adam optimizer with default parameters, including the learning rate. The loss function employed was Binary Cross Entropy with Logits Loss, suitable for binary classification tasks where the output layer lacks an activation function like Sigmoid Sigmoid [25]. The training was conducted over 10 epochs, with training metrics recorded for subsequent evaluation in the results section.

C. Result and Discussion

All models were trained for 10 epochs, during which time and metrics were tracked. This method made it easier to compare the two architectures regarding overall training time, model accuracy per epoch, and model loss per epoch. The performance of the models was then assessed using the test dataset. Monitoring these measures to comprehend model behaviour and guarantee convergence throughout training is essential. Learning curves are useful tools in this context that plot model performance over iterations [39]. Table 2 shows the total model's training summary in terms of time for 10 training iterations.

Table 2. Models' Training Time and Last Saved Epoch

CNN MODEL	*TOTAL*	*LAST SAVED*	*TOTAL TRAINING TIME*
1D	10	7	1min 41.89sec
2D	10	4	4min 11.73sec

The CNN model using a 2D convolutional architecture (CNN2D) took longer to finish the training process over 10 rounds, according to Table 2. On a typical GPU, it finished training in 4 minutes and 11.73 seconds. The entire training time for each model per epoch is shown in Figure 7.

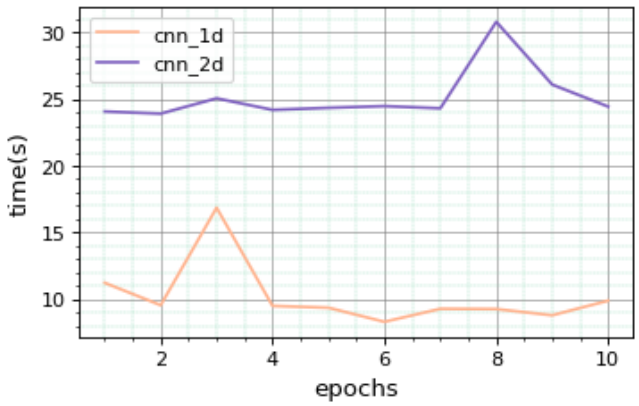

Figure 7. CNN Models' general architecture

Figure 7 shows that, in comparison to the CNN1D model architecture, the CNN2D model architecture took longer to finish each training epoch. The higher computational cost of 2D convolution processes is the reason for this longer training period. Figure 8 shows the model training metrics, such as accuracy and loss trends over epochs, that were captured during the training phase.

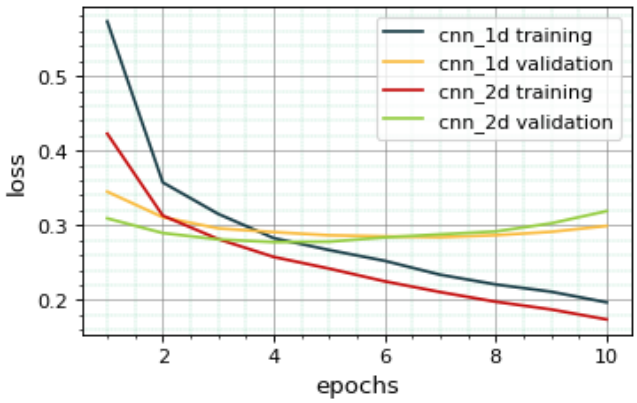

Figure 8. Models' epoch training and validation losses

Effective learning is demonstrated by Figure 8, which shows that the training loss for both models progressively dropped from the first to the last epoch. However, before the models started to overfit around epoch 6, the validation loss decreased over the first 4 epochs. This implies that rather than generalising to previously unknown data, the models began to memorise the training data. The training and validation accuracies over the ten training epochs are shown in Figure 9.

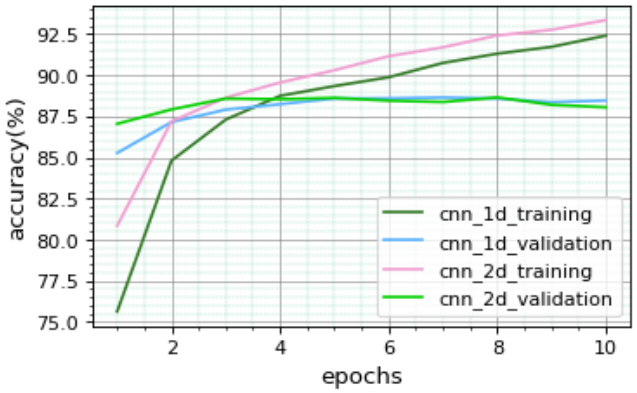

Figure 9. Models' epoch training and validation accuracies

While both models' training accuracies rose quickly from the first epoch, Figure 9 demonstrates that the CNN1D architecture began with less than 80% accuracy. Both models started to overfit the training data as the validation accuracies improved until epoch 4, which suggests a decreased capacity to generalise to new examples.

The best models were saved during model training if the validation loss at that moment was less than the validation loss that had been previously recorded. The model with the lowest validation loss is the best throughout the training phase. This method is known as model checkpointing. By maintaining the state of the model that performs best on the validation data, model checkpointing helps avoid overfitting and preserves the model's capacity for generalisation [40]. Following training, the models were assessed using the test dataset, and the outcomes are shown in Table 3.

Table 3. Best models' testing losses and accuracies

CNN MODEL	*LOSS*	*ACCURACY (%)*
1D	0.290	88.20
2D	0.286	88.41

Table 3 shows that, in terms of test loss and accuracy, the CNN2D model outperformed the CNN1D model, achieving an accuracy of 88.41% and a loss of 0.286. In contrast, the CNN1D model achieved an accuracy of 88.2% and a loss of 0.29 after being evaluated on the test data. Figure 10 shows a confusion matrix plot for the best saved CNN1D model after it has been evaluated on the testing data.

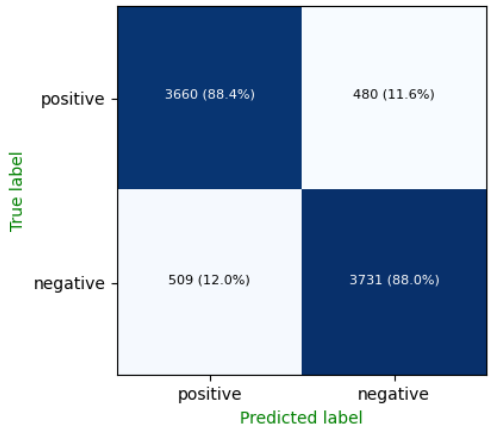

Figure 10. Best CNN 1D model's confusion matrix on the testing dataset

The CNN1D model's confusion matrix based on the test dataset is shown in Figure 10. With a correct classification percentage of 88.0% for negative reviews, the model correctly predicted 88.4% of the total reviews. On the other hand, 11.6% of positive reviews were incorrectly labelled as negative, and 12.0% of negative reviews were incorrectly classified as positive. Figure 11 shows a confusion matrix plot for the best saved CNN2D model after it has been evaluated on the testing data. According to recent research, these findings highlight how well CNN architectures perform sentiment analysis tasks [41].

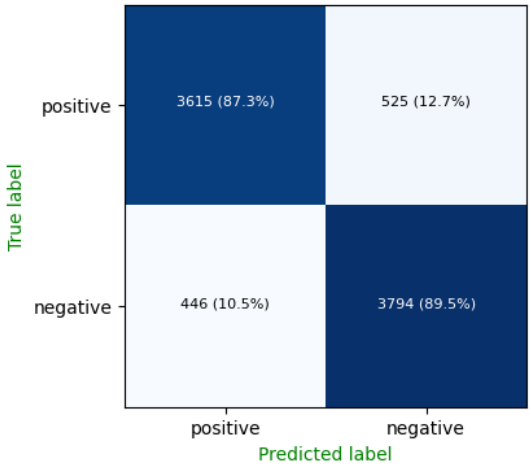

Figure 11. Best CNN 2D model's confusion matrix on the testing dataset

Figure 11 shows the CNN2D model's confusion matrix based on the test dataset. With an accuracy rating of 88.5% for negative reviews, the algorithm accurately predicted 87.3% of all reviews. On the other hand, 12.7% of positive reviews were incorrectly labelled as negative, and 10.5% of negative reviews were incorrectly classified as positive. These results are consistent with recent research that examined CNN architecture performance in sentiment analysis tasks [42]. The CNN1D model's classification report based on the testing dataset is displayed in Figure 12.

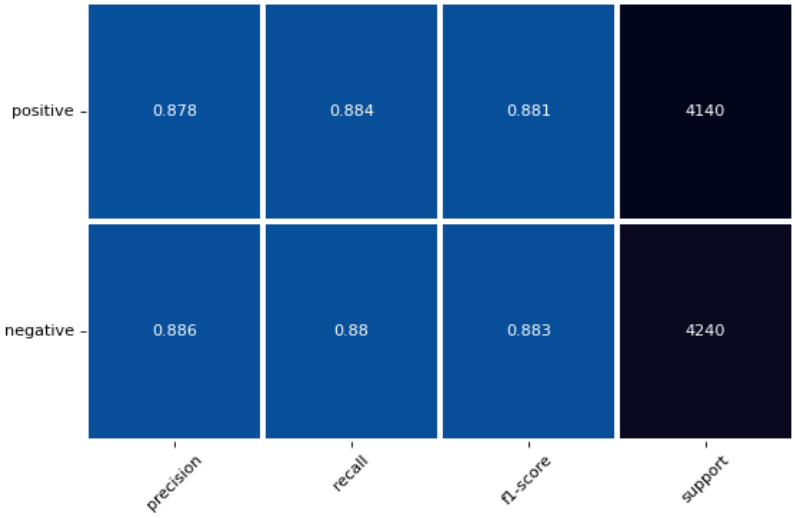

Figure 12. Best CNN 1D model's classification report on the testing dataset

The CNN1D model's classification report, which highlights its performance across important evaluation metrics, is shown in Figure 12. The model produced an excellent overall F1-score with precision of 0.878 for positive reviews and 0.866 for negative reviews, with recall values of 0.884 and 0.88, respectively. The dataset was balanced, as evidenced by the support value 4,140 for both positive and negative evaluations, which shows the number of actual instances per class. This equilibrium lowers the possibility of evaluation bias by guaranteeing that the performance measures are computed on adequate samples [43]. The CNN2D model architecture's classification report is shown in Figure 13.

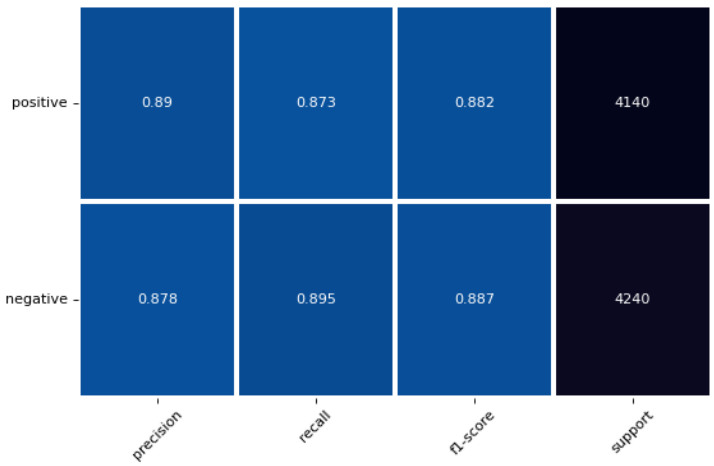

Figure 13. Best CNN 1D model's classification report on the testing dataset

Figure 13 shows the CNN2D model's classification report, which highlights its performance across important evaluation metrics. The model's recall values were 0.873 and 0.895, respectively, and its precision was 0.890 for positive and 0.878 for negative reviews. Strong predictive performance was demonstrated by the total F1-score of 0.882 for positive reviews and 0.887 for negative reviews. The dataset was balanced, as evidenced by the support value 4,140 for both classes, guaranteeing accurate performance assessment across all parameters. According to these results, the CNN2D model showed somewhat greater consistency even though both models performed well, consistent with previous findings in deep learning research.

After training the model, the best model was used to make predictions using the inference subset. Table 4 shows the inference data frame for the first 10 music reviews with their predicted labels.

Table 4. Best models' testing losses and accuracies

Text	***Label***	***Conv1D Prediction***	***Conv1D Probability***	***Conv2D Prediction***	***Conv2D Probability***
We love Spotify. We can find almost anything w…	positive	positive	0.852	positive	0.801
I really do enjoy the amount of music this app…	negative	negative	0.988	negative	0.978
Very good music apps, even though I'm new to us…	positive	positive	0.932	positive	0.930
Why do I pay for premium if I don't even have …	negative	negative	0.967	negative	0.968
This app is useless, unless not being able to …	negative	negative	0.975	negative	0.976
There's some issue in the new version I suppos…	negative	negative	0.999	negative	0.999
An amazing app but if you don't have a sort of…	positive	positive	0.510	positive	0.627
It looks pretty but it won't let me do anythin…	negative	negative	0.934	negative	0.914
This is a great music app, my only problem is …	negative	negative	0.874	negative	0.886
Doing well	positive	positive	0.845	positive	0.867

Table 4 represents prediction DataFrame presents a comparative analysis of sentiment classification results from two different CNN models: CNN1D and CNN2D. Each row in the table corresponds to a text sample, displaying the actual sentiment label (label), the predicted sentiment by CNN1D (conv1d_pred) and CNN2D (conv2d_pred), along with their respective confidence scores (conv1d_probability and conv2d_probability).

From the dataset, most predictions align correctly with the actual labels, demonstrating the effectiveness of both CNN models in sentiment classification. For example, in row 0, both models correctly classify the text as "positive", with CNN1D having a probability of 0.852 and CNN2D at 0.801. However, there are misclassifications, such as in row 6, where CNN1D incorrectly predicts "positive" for a "negative" labelled text with a 0.510 probability, while CNN2D correctly classifies it as "negative" with a 0.627 probability. Similarly, in row 8, both models

incorrectly classify a “positive” text as “negative”, though their probability scores are relatively high.

The probability values indicate the models' confidence in their predictions. Generally, higher probability values close to 1.0 suggest stronger model certainty, as seen in row 5, where CNN1D and CNN2D classify a “negative” review with 0.999 confidence. The prediction variation highlights the differences in model architecture and learning patterns between CNN1D and CNN2D. This aligns with prior research suggesting that 2D convolutional models can leverage spatial relationships better than 1D models, often improving performance in certain classification tasks [25]. However, the misclassification cases suggest that CNN models might still struggle with nuanced language interpretation, a challenge commonly addressed using hybrid deep learning approaches such as CNN-LSTM [24].

The results of comparing one-dimensional (1D) and two-dimensional (2D) CNNs for sentiment analysis of Spotify music evaluations are thoroughly discussed in this section. Model performance is discussed in terms of precision, recall, F1-score, training duration, training accuracy, validation accuracy, testing accuracy, and prediction results.

According to the examination of the two CNN architectures, the CNN2D model fared marginally better in classification accuracy than the CNN1D model. With a loss of 0.290, the CNN1D model's testing accuracy was 88.20%, whereas the CNN2D model's accuracy was 88.41% with a smaller loss of 0.286. In line with findings by Kim and Jeong [4], who highlighted that 2D CNN architectures might more successfully utilize spatial linkages in classification tasks, these results show that the CNN2D model showed superior generalization capabilities when compared to the CNN1D model.

Both models showed an improvement in accuracy and a reduction in loss over epochs during the training phase. However, around the sixth epoch, overfitting was noticed, as seen in Figures 8 and 9, and training accuracy kept getting better while validation accuracy started to plateau. This implies that instead of generalising to unknown examples, a common problem in deep learning models, both models began memorising training data [10].

More information on the classification performance of the CNN1D and CNN2D models may be found in their confusion matrices. While the CNN2D model achieved a slightly higher classification accuracy, successfully categorising 87.3% of positive reviews and 88.5% of negative reviews, the CNN1D model correctly classified 88.4% of positive reviews and 88.0% of negative reviews. CNN2D misclassified somewhat more positive reviews than CNN1D, although having a stronger recall for negative reviews, according to both models' false positive and false negative rates. This result is consistent with an earlier study by Yildirim [42], which indicates that 2D CNNs may be better at capturing spatial information but still have trouble classifying text nuancedly.

The precision, recall, and F1-score of the classification reports for the two models were similar. With recall values of 0.884 and 0.880, respectively, the CNN1D model obtained a precision of 0.878 for positive and 0.866 for negative reviews. With recall values of 0.873 and 0.895, respectively, and precision of 0.890 for positive and 0.878 for negative reviews, the CNN2D model performed

marginally better than the CNN1D model. Better sensitivity to detecting negative attitudes is indicated by CNN2D's stronger recall for unfavourable reviews. Diwan and Tembhurne's [16] findings, which pointed out that CNN-based architectures frequently have significant predictive power in sentiment classification tasks, are consistent with this.

The two models' training times differed significantly from one another. The CNN2D model took 4 minutes and 11.73 seconds to train, while the CNN1D model took 1 minute and 41.89 seconds. According to Park, Lee, and Sim [5], the extra computational complexity involved in two-dimensional convolution processes is the reason behind CNN2D's longer training time. CNN2D showed marginally improved performance despite the increased computational cost, suggesting a trade-off between accuracy and economy.

With a few misclassifications, the inference results showed that both models did well in sentiment categorisation. For example, CNN2D properly categorised a negative review with a probability of 0.627, whereas CNN1D misclassified it as positive with a chance of 0.510. These variations in prediction confidence imply that CNN2D was somewhat more adept at understanding context-dependent sentiment subtleties. The minor increase in CNN2D'S prediction accuracy may be explained by the fact that 2D CNNS are better at capturing intricate correlations between textual elements than 1D CNNS [42].

D. Conclusion

The CNN1D and CNN2D architectures for sentiment analysis of Spotify music evaluations were compared in this work. The findings show that CNN2D performs somewhat better than CNN1D in terms of accuracy and recall but at the cost of increased training time and computational complexity. CNN2D has better feature extraction capabilities, especially for complicated sentiment patterns, although CNN1D is still a good choice for efficiency. To strike a compromise between accuracy and computing economy, future studies should investigate hybrid models and optimisation strategies. This work offers important new information about sentiment analysis using deep learning, especially in music review datasets.

Future research could examine hybrid architectures like CNN-LSTMs, which, as proposed by Islam et al. [17], combine the sequential memory capacities of LSTMs with the spatial feature extraction of CNNs. Model performance may also be improved by adjusting hyperparameters, expanding the dataset, and utilising transfer learning strategies.

E. Acknowledgment

The authors would like to acknowledge the creator of the Spotify User Reviews dataset available on Kaggle, which provided the foundation for this research. Their contribution to open-source data sharing has significantly aided the advancement of sentiment analysis methodologies. Additionally, we extend our gratitude to the deep learning research community for continuously developing innovative architectures that enhance text classification tasks.

References

[1] J. F. Sánchez-Rada and C. A. Iglesias, "Social context in sentiment analysis: Formal definition, overview of current trends and framework for comparison," *Information Fusion*, vol. 52, pp. 344–356, Dec. 2019, doi: 10.1016/j.inffus.2019.05.003.

[2] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," Artif Intell Rev, vol. 55, no. 7, pp. 5731–5780, Oct. 2022, doi: 10.1007/s10462-022-10144-1.

[3] G. M. Biancofiore, T. Di Noia, E. Di Sciascio, F. Narducci, and P. Pastore, "Aspect Based Sentiment Analysis in Music: a case study with Spotify," in Proceedings of the ACM Symposium on Applied Computing, Association for Computing Machinery, Apr. 2022, pp. 696–703. doi: 10.1145/3477314.3507092.

[4] H. Kim and Y. S. Jeong, "Sentiment classification using Convolutional Neural Networks," Applied Sciences (Switzerland), vol. 9, no. 11, Jun. 2019, doi: 10.3390/app9112347.

[5] J. Park, J. Lee, and D. Sim, "Low-complexity CNN with 1D and 2D filters for super-resolution," in Journal of Real-Time Image Processing, Springer Science and Business Media Deutschland GmbH, Dec. 2020, pp. 2065–2076. doi: 10.1007/s11554-020-01019-1.

[6] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," Mech Syst Signal Process, vol. 151, Apr. 2021, doi: 10.1016/j.ymssp.2020.107398.

[7] X. Hu, J. S. Downie, and A. F. Ehmann, "LYRIC TEXT MINING IN MUSIC MOOD CLASSIFICATION," 2009. [Online]. Available: http://en.wikipedia.org/wiki/MoodLogic

[8] A. Aljanaki, F. Wiering, and R. C. Veltkamp, "Studying emotion induced by music through a crowdsourcing game," Inf Process Manag, vol. 52, no. 1, pp. 115–128, Jan. 2016, doi: 10.1016/j.ipm.2015.03.004.

[9] B. Liu, "Sentiment Analysis and Opinion Mining," Morgan & Claypool Publishers, 2012.

[10] L. Yue, W. Chen, X. Li, W. Zuo, and M. Yin, "A survey of sentiment analysis in social media," Knowl Inf Syst, vol. 60, no. 2, pp. 617–663, Aug. 2019, doi: 10.1007/s10115-018-1236-4.

[11] P. C. Tetlock, M. Saar-Tsechansky, and S. Macskassy, "More Than Words: Quantifying Language to Measure Firms' Fundamentals," Journal of Finance, vol. 63, no. 3, pp. 1437–1467, 2008, doi: 10.1111/j.1540-6261.2008.01362.x.

[12] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe, "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment," 2010. [Online]. Available: www.aaai.org

[13] E. Cambria, B. Schuller, Y. Xia, and C. Havasi, "New Avenues in Opinion Mining and Sentiment Analysis." [Online]. Available: http://converseon.com

[14] C. Matobobo and F. Bankole, "Evaluating eWOM in Social Media: Religious Leaders vs Religious Organizations: Functionality Approach," in UK Academy for Information Systems Conference Proceedings 2021, 2021.

[15] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," 2014. [Online]. Available: http://sentic.net/
[16] T. Diwan and J. V. Tembhurne, "Sentiment analysis: a convolutional neural networks perspective," Multimed Tools Appl, vol. 81, no. 30, pp. 44405–44429, Dec. 2022, doi: 10.1007/s11042-021-11759-2.
[17] M. S. Islam et al., "'Challenges and future in deep learning for sentiment analysis: a comprehensive review and a proposed novel hybrid approach,'" Artif Intell Rev, vol. 57, no. 3, Mar. 2024, doi: 10.1007/s10462-023-10651-9.
[18] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," IEEE Trans Neural Netw Learn Syst, vol. 28, no. 10, pp. 2222–2232, Oct. 2017, doi: 10.1109/TNNLS.2016.2582924.
[19] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," Dec. 2014, [Online]. Available: http://arxiv.org/abs/1412.3555
[20] Y. Kim, "Convolutional Neural Networks for Sentence Classification," Aug. 2014, [Online]. Available: http://arxiv.org/abs/1408.5882
[21] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," Wiley Interdiscip Rev Data Min Knowl Discov, vol. 8, no. 4, Jul. 2018, doi: 10.1002/widm.1253.
[22] A. Oramas, S. Oramas, L. Espinosa-Anke, A. Lawlor, X. Serra, and H. Saggion, "Exploring customer reviews for music genre classification and evolutionary studies," in The 17th International Society for Music Information Retrieval Conference (ISMIR 2016), 2016. [Online]. Available: http://hdl.handle.net/10197/7975
[23] O. Müller, I. Junglas, J. Vom Brocke, and S. Debortoli, "Utilizing big data analytics for information systems research: Challenges, promises and guidelines," European Journal of Information Systems, vol. 25, no. 4, pp. 289–302, Jul. 2016, doi: 10.1057/ejis.2016.2.
[24] A. Kim, "Spotify User Reviews," Kaggle. Accessed: Nov. 07, 2024. [Online]. Available: https://www.kaggle.com/datasets/alexandrakim2201/spotify-dataset
[25] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, no. 4. Cambridge: MIT Press, 2016. doi: 10.4258/hir.2016.22.4.351.
[26] Y. Yang and Z. Xu, "Rethinking the Value of Labels for Improving Class-Imbalanced Learning," in 34th Conference on Neural Information Processing Systems (NeurIPS 2020), 2020, pp. 1–12. [Online]. Available: https://github.com/YyzHarry/imbalanced-semi-self.
[27] Y. Huang, B. Giledereli, A. Köksal, A. Özgür, and E. Ozkirimli, "Balancing Methods for Multi-label Text Classification with Long-Tailed Class Distribution," arXiv preprint arXiv:2109.04712, Sep. 2021, [Online]. Available: http://arxiv.org/abs/2109.04712
[28] J. Zolfaghari Bengar, J. Van De Weijer, L. L. Fuentes, and B. Raducanu, "Class-Balanced Active Learning for Image Classification," in Proceedings of the IEEE/CVF winter conference on applications of computer vision, 2022, pp. 1536–1545. [Online]. Available: https://github.com/Javadzb/

[29] J. Brownlee, Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python. 2018.
[30] Dan. Jurafsky and J. H. . Martin, Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition. Prentice Hall, 2021.
[31] X. Zhang, J. Zhao, and Y. LeCun, "Character-level Convolutional Networks for Text Classification," in Advances in Neural Information Processing Systems 28 (NIPS 2015), Feb. 2015. [Online]. Available: http://arxiv.org/abs/1502.01710
[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv:1810.04805v2, Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.04805
[33] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," arXiv preprint arXiv:1301.3781, Jan. 2013, [Online]. Available: http://arxiv.org/abs/1301.3781
[34] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in Proceedings ofthe 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.
[35] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," in Proceedings ofthe 56th Annual Meeting of the Association for Computational Linguistics, 2018, pp. 328–339. [Online]. Available: http://nlp.fast.ai/ulmfit.
[36] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," 2014.
[37] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in Proceedings of the thirteenth international conference on artificial intelligence and statistics, 2010, pp. 249–256. [Online]. Available: http://www.iro.umontreal.
[38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.
[39] F. Mohr and J. N. van Rijn, "Learning Curves for Decision Making in Supervised Machine Learning: A Survey," Mach Learn, vol. 113, no. 11, pp. 1–65, Jan. 2021, doi: 10.1007/s10994-024-06619-7.
[40] H. Li, G. K. Rajbahadur, D. Lin, C. P. Bezemer, and Z. M. Jiang, "Keeping Deep Learning Models in Check: A History-Based Approach to Mitigate Overfitting," IEEE Access, vol. 12, pp. 70676–70689, 2024, doi: 10.1109/ACCESS.2024.3402543.
[41] Y. Xie and R. C. Raga Jr, "Convolutional Neural Networks for Sentiment Analysis on Weibo Data: A Natural Language Processing Approach," arXiv preprint arXiv:2307.06540, pp. 1–5, 2024.
[42] S. Yildirim and Y. Santur, "Comparing the Performance of Deep Learning Architectures for Sentiment Analysis," International Journal of Advanced Natural Sciences and Engineering Researches, vol. 4, no. 4, pp. 272–278, 2024, [Online]. Available: https://as-proceeding.com/index.php/ijanser
[43] V. D. Derbentsev, V. S. Bezkorovainyi, A. V Matviychuk, O. M. Pomazun, A. V Hrabariev, and A. M. Hostryk, "A comparative study of deep learning models

for sentiment analysis of social media texts," in M3E2-MLPEED, 2023, pp. 168–188.