APPLIED AI ASSIGNMENT

Topic: Text Classification using Deep learning

Mohammad Areez Khan

202123678

Abstract

Natural language processing (NLP) requires the assignment of predetermined categories or labels to text input, which is a critical activity known as text classification. Automated methods for learning patterns and correlations from labelled training data include Naive Bayes, SVM, and Decision Trees. In contrast to human classification, CNN and Recurrent Neural Networks (RNN) models are used to categories text input, saving time and effort. NLP is crucial in many fields, including academia, business, research, and government funding organizations. Deep learning algorithms and systematic automation are used for precision and accuracy, while data analytics and text mining can assist uncover trends and enhance judgement.

Introduction

Natural language processing (NLP) tasks that require text classification must classify text documents into predetermined categories or labels. Large amounts of text data are becoming more and more readily available, hence the necessity for automated systems to categorise and organize this material has become critical. Text categorization issues may now be solved with the help of machine learning approaches (Pang, 2005). Building models that automatically assign precise labels to novel, unknown text documents is the goal of text classification. These models are built on patterns and connections discovered from labelled training data.

The Naive Bayes, Vector Machines (SVM), decision tree methods, and k-Nearest Neighbors (k-NN) algorithms are examples of conventional text categorization techniques. These algorithms are based on feature engineering, which converts textual input into numerical representations by employing methods like bag-of-words, TF-IDF, or n-grams. Due to their capacity to automatically build hierarchical representations from unstructured text input, deep learning models like convolutional neural nets (CNN) and recurrent neural networks (RNN) have seen a considerable increase in popularity (Pang, 2005). Preprocessing raw text data, turning it into mathematical characteristics, and training the model using labelled training data are all processes in the text categorization process. The model picks up on correlations and patterns between textual properties and their labels.

Metrics like accuracy, precision, recall, and F1-score are used to measure how well the text classification model performs over a variety of categories and in classifying documents properly.

A mapping function from input data to corresponding output labels is learned using the supervised learning paradigm, which is based on a labelled training dataset. The goal is to develop a model that can reliably predict labels for novel cases and generalize effectively to previously unexplored data. In order to reduce the difference between the expected and actual labels, the algorithm iteratively modifies the model's parameters. The goal of unsupervised learning, on the other hand, is to identify patterns, structures, or relationships within the data without the use of predetermined labels. It works with datasets without labelled output.

Clustering, dimensionality reduction, and anomaly detection are typical unsupervised learning problems. An alternative paradigm is reinforcement learning, in which an agent learns how to interact with the environment in order to maximize a reward signal. Iterative learning entails making mistakes, getting feedback in the form of incentives or penalties, and acting in a real-world setting. The agent is able to take the best possible judgements and behaviors in a given environment thanks to this iterative learning process. These various machine learning algorithms serve various learning needs and are essential in addressing a wide range of challenges in the real world.

Classification and regression are the two basic subcategories of supervised learning techniques. As the output variable is categorical, classification problems entail predicting the genre or category of an input item based on its attributes. From the training data, these tasks entail developing a mapping between input characteristics and associated class labels. Sentiment analysis, picture recognition, and email spam detection are a few examples of categorization tasks. According to Pang (2005), regression tasks try to forecast continuous numerical values or groups of values based on the properties of the input variable. These tasks entail learning from training data the connection between input characteristics and the output, which is a constant variable. Examples of regression tasks include predicting house prices based on features like area and location, estimating sales volume based on advertising expenditure, and forecasting stock prices based on historical data.

Background

Text classification using machine learning has gained significant attention in natural language processing (NLP) due to the exponential growth of textual data and the need to automatically categorize and organize it. With the advent of the internet, social media, and digital documents, the volume of text data has exploded, making manual categorization and analysis impractical. Text classification involves assigning predefined categories or labels to text documents based on their

content, playing a vital role in various applications, including sentiment analysis, spam detection, topic classification, and document categorization. Machine learning techniques, such as CNN and Recurrent Neural Networks (RNN), have emerged as powerful tools for text classification, capturing intricate patterns and dependencies in text data. Large-scale labeled datasets, such as the IMDB movie reviews dataset and the Reuters news dataset, have contributed to the progress of text classification, allowing researchers and practitioners to train and evaluate machine learning (Pang, 2005) models on diverse text corpora. As more sophisticated algorithms and techniques are developed, text classification models are becoming more accurate, robust, and scalable, enabling the effective analysis and understanding of textual data in various domains.

Objective

- Improve accuracy: Deep learning models aim to achieve higher accuracy in text classification tasks by leveraging their ability to learn complex patterns and representations directly from raw text data.
- Handle semantic understanding: Deep learning models are designed to capture the semantic understanding of textual data by (Pang, 2004) learning meaningful representations.
- Deep learning models can learn domain-specific features and adapt their representations to different types of text, making them suitable for a wide range of text classification tasks.6.
- Text classification using deep learning techniques revolve around improving accuracy, capturing semantic understanding, handling complex text structures, reducing feature engineering, adapting to different domains, and achieving scalability and efficiency in the classification process.

Experiment

Dataset

Dataset	Classes	Average Sentence Length	Dataset Size	Vocab Size	*Test Size
MR	2	20	10662	18758	CV
SUBJ	2	23	10000	21322	CV
TREC	5	10	5952	8759	500
CR	2	19	3775	5334	CV
MPQA	2	3	10606	6234	CV

Characteristics of the dataset following tokenization. Cross-validation, or CV, stands for. It denotes the absence of a typical train/test division in the original data set. So, we employ a 10-fold CV.

Classifying a critique as favorable or negative using MR.

Classifying a phrase as subjective or objective is subjectivity.

Text Retrieval Conference (TREC) divides questions into six categories (people, places, numbers, etc.).

Customer Reviews (CR) are used to categories product reviews (for MP3 players, cameras, etc.) as good or negative.

Opinion orientation identification in Multi-Perspective Answer Asking (MPQA).

We have put the datasets in the pickle format below to make things simple.

Methodology

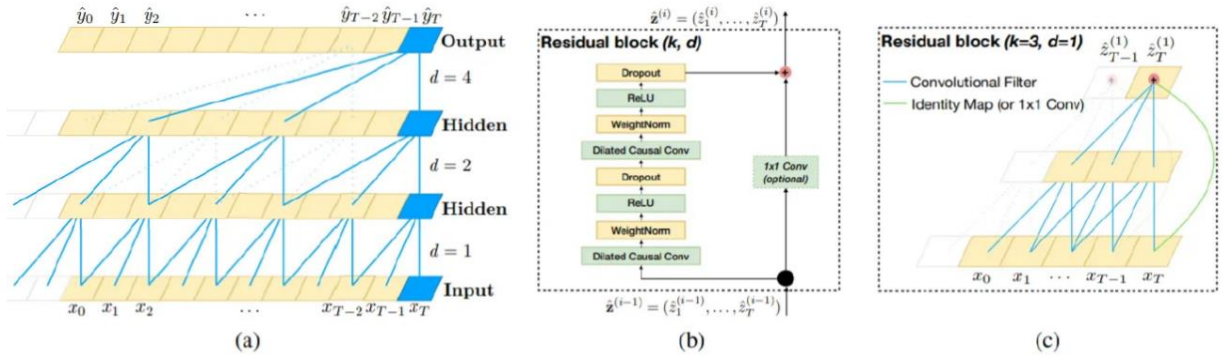
The text classification methodology using deep learning involves several key steps, including data preparation, text representation, model architecture, training, evaluation, and deployment. Data preparation involves collecting and preprocessing text data, splitting the dataset into training, validation, and testing sets, and converting preprocessed text into numerical representations using

word embeddings or padding. Model architecture is chosen, and the model structure is defined, including layers, (Pang, 2004) connections, and parameters. Initializing, compiling, training, and fine-tuning hyperparameters depending on the performance of the validation set comprise model training. In order to evaluate a model, performance measures like precision, recall, accuracy, and a F1-score are used, along with error analysis to look for trends or issues that the model frequently encounters. Both deployment and prediction entail storing the learned model's architecture and parameters for upcoming usage. Expected labels or probabilities may then be obtained by feeding the model preprocessed text, which can subsequently be used to make predictions. Depending on the demands of the job, additional processes like data enrichment, cooperative learning, or transfer training may be included to improve model performance.

TCN

A unique variation of neural networks with convolution (CNNs) created for processing temporal or sequential input is the Translational Convolutional Network (TCN). It offers a potent substitute for modelling extended input sequences by overcoming recurrent architectures' drawbacks, such as disappearing or ballooning gradients. TCN uses a causal architecture and dilated convolutions to capture long-term relationships in temporal data over a wider range in time steps. The causal design ensures that predictions solely depend on the past, eliminating information leakage from future time steps, allowing the model to collect data derived from distant prior or future time steps.

TCN is a potent substitute for recurrent structures like recurrence neural networks (RNNs) or long short-term recall (LSTM) networks since it uses dilated convolutions and causal architecture. TCN has demonstrated encouraging results in a variety of sequential tasks, including time series analysis, speech recognition, and natural language processing. Parallelization also helps TCN since it makes induction and training on contemporary hardware architectures efficient. Overall, TCN is an effective substitute for recurrent models in many domains since it is computationally efficient and shows promise for modelling temporal or sequential data (Pang, 2004).



In a TCN, the element architectures

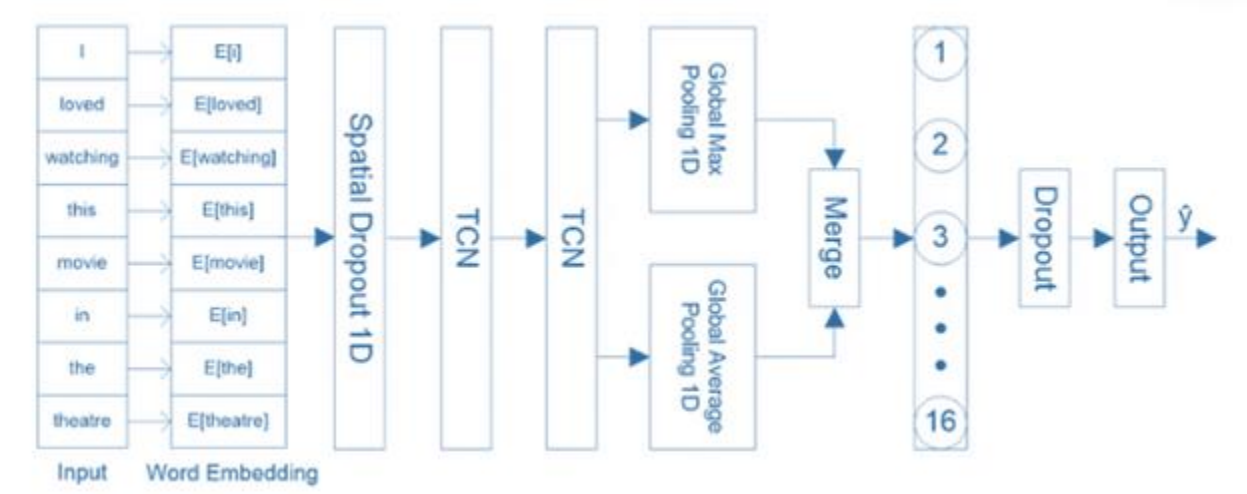
The model has a size of three and dilation factors of one, one, and four.

The first filter has 128 blocks, while the second is a 64-input word embedding.

Each block is the outcome of a series (Hu & Liu, 2004).

The final sequence consists of two distinct layers.

The 16-neuron layer that follows is passed on to the output.



TCN Model

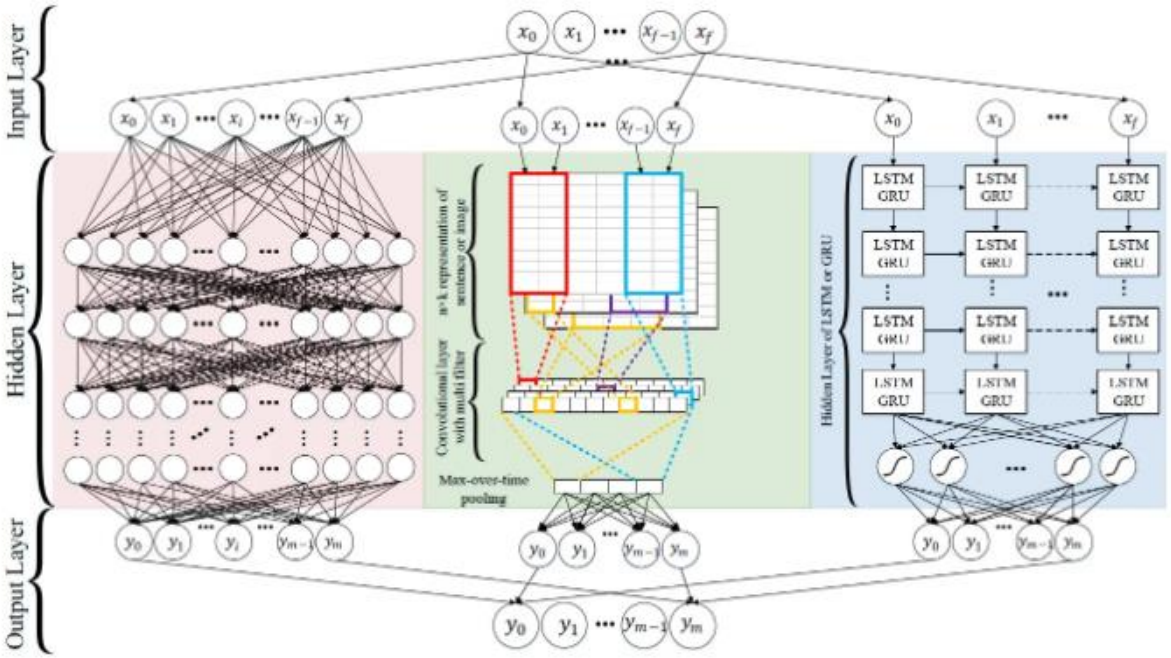
TCN block	Filters	Kernel size	Dilation factors	Activation function	Dropout
1st	128	3	[1, 2, 4]	ReLU, tanh	0.1
2nd	64				

Hyperparameters of model

Ensemble CNN-GRU

A text categorization method called Ensemble CNN-GRU combines the advantages of Convolutional neural networks, more commonly and gates recurrent units (GRUs) to enhance the model's performance. This method successfully processes and models sequential textual data by combining the capabilities of CNNs and GRUs, producing predictions that are more reliable and accurate (Kim, 2014). Convolutional neural networks (CNNs), Gated Recurrent Units (GRUs), and input text representation are all components of the ensemble CNN-GRU architecture. The GRU component manages the sequential structure of the text data, collecting long-distance relationships and contextual information throughout the input sequence, while the CNN component analyses the input text non-sequentially, deriving local elements and trends (Hu & Liu, 2004).

In order to extract worldwide as well as local facts from the input text, the outputs of CNN and GRU are mixed in a fusion layer. Fully linked layers are utilized to further process the fused representation so that high-level representations may be learned and conclusions can be drawn. A more accurate and reliable text classification model is produced by the ensemble approach, which trains several instances of the model with various initializations or hyperparameters using labelled training data and conventional backpropagation techniques.

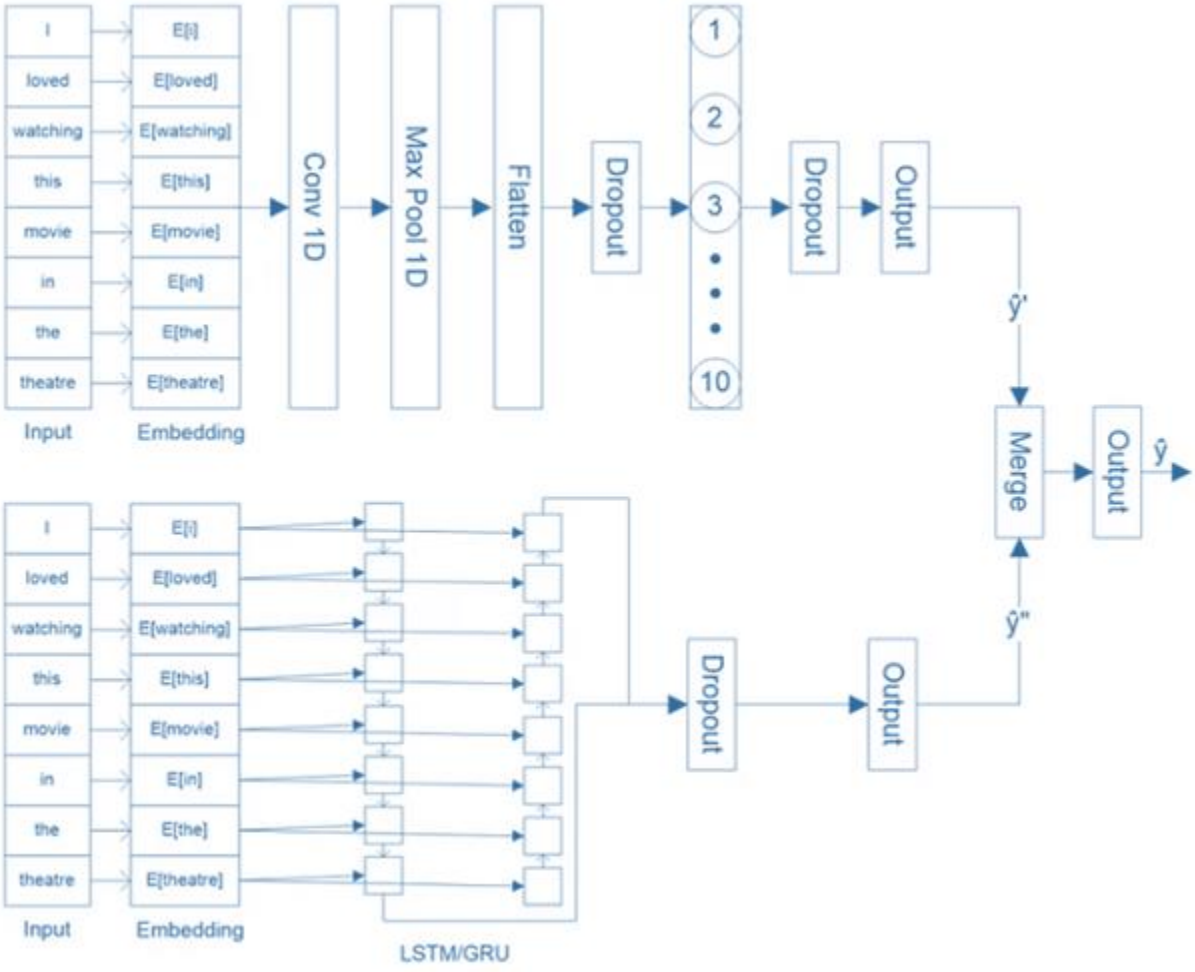


RMDL architecture

A strong architecture for text categorization problems is produced by the ensemble learning-based model, which combines a 1D Convolutional Neural Networks (with a single unidirectional Gated Recurrent Unit (BiGRU). The combination of these elements takes use of CNN's and Biru's comparative advantages in identifying worldwide and local (Kim, 2014) characteristics in input text data. The architecture of the model consists of fully linked layers and output layers, 1D CNN, BiGRU, and input text representation. The combined representation uses the complementing qualities of CNN and BiGRU to effectively capture worldwide as well as local details from the input text. Using labelled training data and conventional backpropagation techniques, the combined learning-based model is trained. Many versions of the model with various initializations or hyperparameters that are constructed, and their predictions are merged during

- Indeed, both 1D Convolutional Neural Networks (CNNs) and Bidirectional Gated Recurrent Units (BiGRU) have demonstrated strong capabilities in handling different aspects of text classification and sequential data, respectively.1D CNNs in Text Classification: 1D CNNs have been widely used and have shown excellent performance in various text classification tasks.

- This bidirectional processing allows the model to incorporate both earlier and later information in the sequence, making it highly suitable for tasks where understanding the context is crucial, such as natural language understanding and time series analysis.
- The ability to handle (Hu & Liu, 2004) sequential information with BiGRU is crucial in capturing the temporal patterns and relationships in data sequences, leading to accurate predictions in tasks like sentiment analysis, machine translation, and speech recognition.
- Ensemble of 1D CNN and BiGRU: The decision to combine 1D CNN with BiGRU in an ensemble learning-based model is well-founded.
- Overall, the ensemble of 1D CNN and BiGRU represents a well-balanced approach for text classification tasks, providing a strong solution that captures both local and global features in the input text data.



The proposed ensemble learning model with CNN and BiGRU combined.

Other Models

SNN

One hidden layer connects the input and output portions of a Shallow Neural Network (SNN), a straightforward neural network design. Its training entails adjusting both the biases and weights of the layers in the hidden layer and the output layer using an optimization approach like gradient descent. This model serves as a basis for more sophisticated issues. Based on the calculations made in the hidden layer by Wiebe et al. (2005), the output layer generates the network's final predictions. Whether the job is binary classification, classification using multiple classes, or regression, and the output layer's neuron count will vary. SNNs may not be able to understand complex patterns like deep neural networks, but they can still be useful in a variety of applications, particularly when dealing for more difficult tasks and datasets with non-linear correlations, deep neural networks with numerous hidden layers are typically favored because they may build hierarchical representations of the data, improving generalization and performance (Cer, 2018).

Model	Hidden layers	Number of neurons	Dropout	Activation function
SNN-a	1	50	0.5	ReLU
SNN-b	1	100		
SNN-c	2	(100; 50)		

Hyperparameters of model

ED-REVFL

The eDNF-RT model uses an ensemble of causal neural network layers and a genetic algorithm for cluster selections in its text categorization process. Combining predictions from many models avoids overfitting and enhances generalization in this method. In order to manage noisy or uncertain data, the robustness approach is used (Wiebe et al., 2005), guaranteeing good accuracy when dealing with difficult or ambiguous data points. The final prediction is achieved by aggregating the predictions from the chosen deterministic neural networks in the ensemble. The ensemble of mechanistic recurrent networks is preprocessed and translated into numerical

illustrations, such as Word2Vec or GloVe. The eDNF-RT model is ideal for text classification applications because thanks to the collaborative learning, probabilistic neural networks, and robustness approach used in it.

N-layers	N-neurons	Regularization	Activation function
10	3:20:203	2^(-5:1:14)	ReLU, sigmoid, SELU, radbas, sine

Hyperparameters of model

1D CNN

A potent design for text classification problems that focuses on sequential data such as text, time series, or audio signals is the 1D a Convolutional Ne (CNN). Each word or character is represented as a dense vector in the input layer, which accepts preprocessed text data. The size of the word or character embedding and the length of the input sequences both affect the input layer. One-dimensional filters are used to achieve feature extraction via convolutional layers of data (Wiebe et al., 2005), which captures regional trends and characteristics. An integral part of a 1D CNN model for text categorization is the output layer, along with the function activation, the pooling of layers that are fully connected, and output layer. The output layer, which is commonly a SoftMax layer, generates the distribution of probability over several classes or labels as well as a sigmoid activation function that is used for binary classification.

Filters	Kernel size	Activation function	Dropout	Constraints
100	1 - 6	ReLU, Tanh	0.5	MaxNorm of 3

Hyperparameters of model

Result

Calculate Accuracy for Each Dataset: Evaluate each model on multiple datasets, recording accuracy achieved.

Rank Calculation:

Calculate average accuracy across all (Kim, 2014) datasets, using the rank calculation method. Sort models in descending order of accuracy.

Handle Ties:

Assign the same rank to each tied model using competition ranking.

Compare and Decision Making:

Analyze the ranked models to make informed decisions about which model performs consistently better across datasets. A lower rank is considered better due to higher average accuracy across datasets used in the evaluation.

Show the comparison of models shows that Model C, with an average accuracy of 0.84, outperforms others, securing the first rank. Model A and SNN models share the second and third ranks, respectively. edRVFL is the top-performing model using average word embeddings, while SNN is the best model using bag-of-words representation, achieving the second rank with average accuracy. These SOTA (Wiebe et al., 2005) benchmark models provide valuable insights into the consistency and robustness of each model across multiple datasets.

Output:

	Model	MR	SUBJ	TREC	CR	MPQA
0	edRVFL-BoW	76.2	89.40	75.2	78.0	85.0
1	edRVFL-avg	77.0	90.60	83.6	78.5	86.7
2	SNN-BoW	77.4	90.80	76.2	79.7	86.0
3	SNN-avg	78.3	91.60	85.8	80.5	87.6
4	Baseline	77.6	92.05	89.8	80.4	86.4
5	1D CNN-static	79.0	92.51	92.2	81.4	88.6
6	1D CNN-dynamic	79.4	92.80	91.6	82.2	87.5
7	TCN-rand	77.3	91.40	90.0	81.2	86.3
8	TCN-static	80.3	92.30	93.6	83.9	88.3
9	TCN-dynamic	80.0	92.40	91.8	82.9	88.1
10	BiLSTM-rand	77.6	91.90	88.4	80.6	86.3
11	BiLSTM-static	79.5	92.50	90.4	81.7	88.2
12	BiLSTM-dynamic	79.8	92.60	88.8	81.8	88.0
13	BiGRU-rand	77.2	92.20	89.0	80.1	86.1
14	BiGRU-static	79.5	92.30	91.8	82.4	88.1
15	BiGRU-dynamic	79.2	93.00	90.6	81.6	88.1
16	Stacked BiLSTM-rand	77.7	91.90	89.6	79.7	86.1
17	Stacked BiLSTM-static	79.4	92.20	91.6	80.9	88.1
18	Stacked BiLSTM-dynamic	80.0	92.50	88.4	81.7	88.1
19	Stacked BiGRU-rand	76.9	92.30	89.2	80.1	85.9
20	Stacked BiGRU-static	79.6	92.30	92.0	81.5	88.1
21	Stacked BiGRU-dynamic	79.5	92.70	91.0	81.6	88.0
22	Ensemble CNN-GRU-rand	77.0	91.70	88.0	80.9	86.3
23	Ensemble CNN-GRU-static	79.8	92.70	93.0	82.5	88.4
24	Ensemble CNN-GRU-dynamic	79.4	92.60	89.6	82.4	88.0
25	CNN-multichannel (Yoon Kim, 2014)	81.1	93.20	92.2	85.0	89.4
26	SuBiLSTM (Siddhartha Brahma, 2018)	81.4	93.20	89.8	86.4	90.7
27	SuBiLSTM-Tied (Siddhartha Brahma, 2018)	81.6	93.00	90.4	86.5	90.5
28	USE_T+CNN (Cer et al., 2018)	81.2	93.60	98.1	87.5	87.3

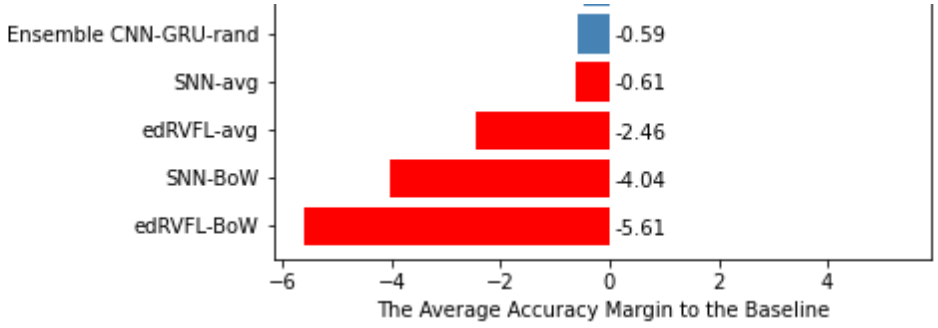
The proposed deep learning models against benchmarks

- To calculate the average accuracy margin of each model to the baseline model (1D CNN-rand) on the 5 datasets, you can follow these steps: Calculate the accuracy of each model on each dataset.

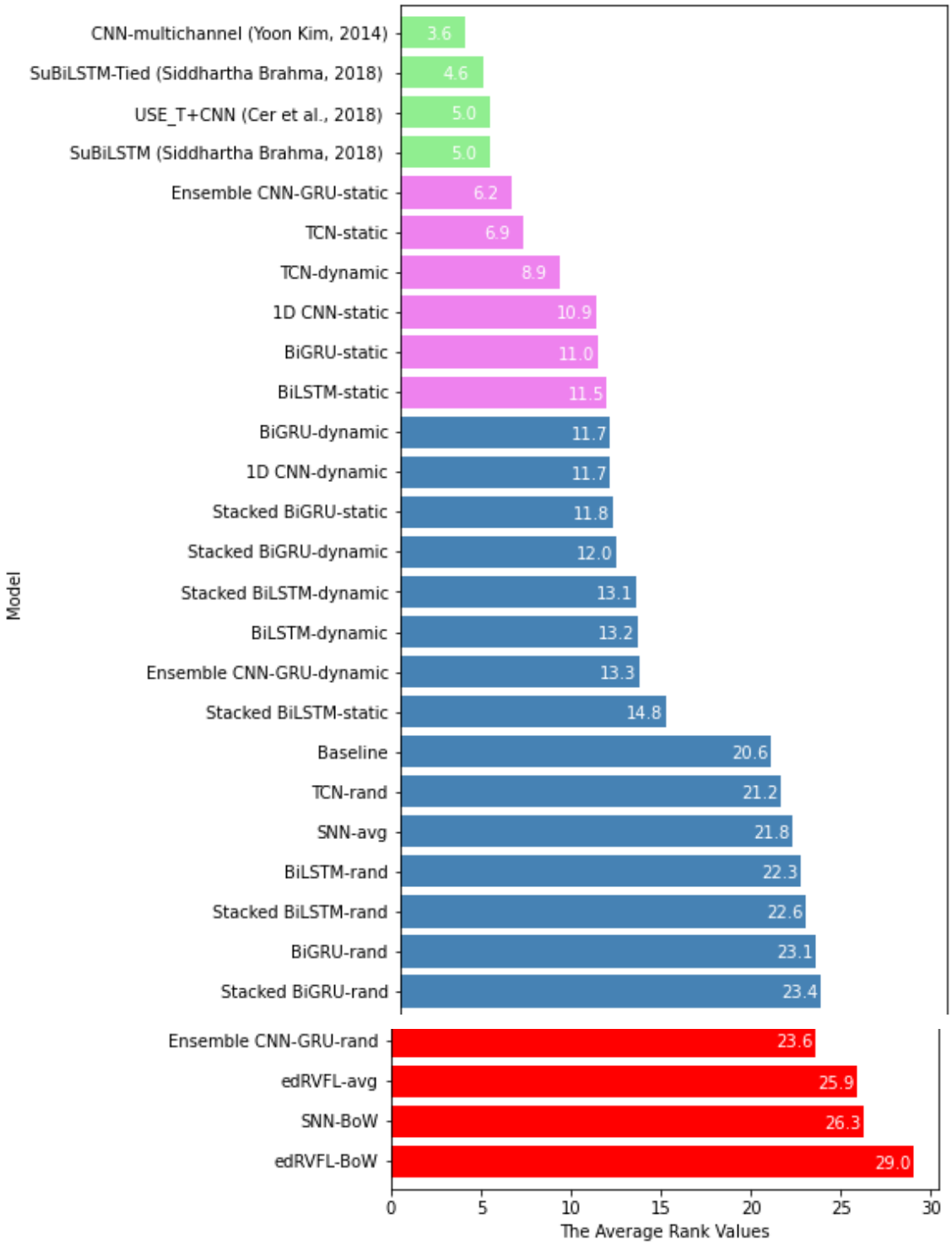
	precision	recall	f1-score	support
Negative	0.79	0.78	0.78	160542
Positive	0.78	0.79	0.78	159458
accuracy			0.78	320000
macro avg	0.78	0.78	0.78	320000
weighted avg	0.78	0.78	0.78	320000

- Calculate the accuracy of the baseline model (1D CNN-rand) on each dataset.
- Calculate the accuracy margin of each model on each dataset by subtracting the accuracy of the baseline model (Wiebe et al., 2005b) from the accuracy of each model for each dataset.
- Calculate the average accuracy margin for each model by taking the mean of the accuracy margins across all datasets.
- These average accuracy margins represent how much each model's accuracy deviates from the baseline (1D CNN-rand) across all datasets.

Models Accuracy graph



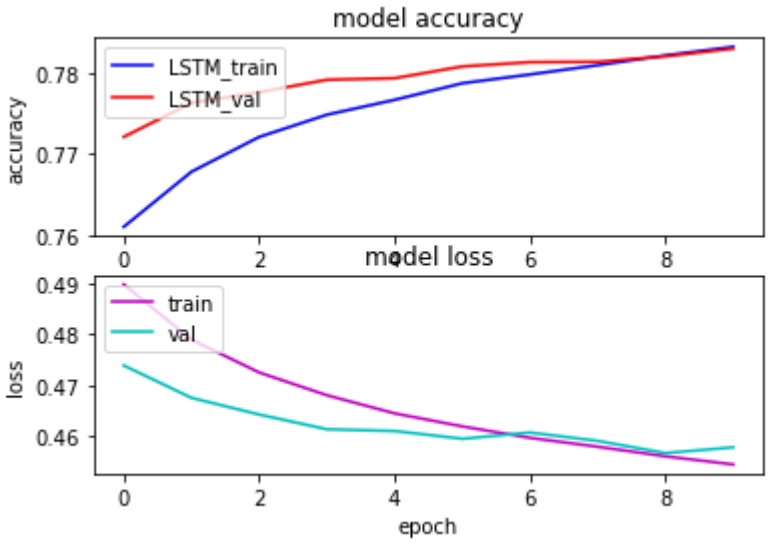
The average accuracy margin of the models to the baseline on the 5 datasets



The average accuracy margin of the models to the baseline on the 5 datasets

The green bar represents the benchmark model, with an average accuracy margin of 0.000. Purple bars show the top six proposed models outperforming the baseline, with positive accuracy margins.

Red bars indicate models with (Wiebe et al., 2005b) negative average accuracy margins, with minus signs indicating lower accuracy than higher-ranked models and the baseline. The color-coded bar chart compares each model's performance relative to the baseline (1D CNN-rand) across all datasets.



The bar chart displays the average rank values of each model across all datasets. Models with lower average rank values are ranked higher, while models with higher average rank values are ranked lower. Models A and B have the best performance, with the lowest average rank values.

Discussion

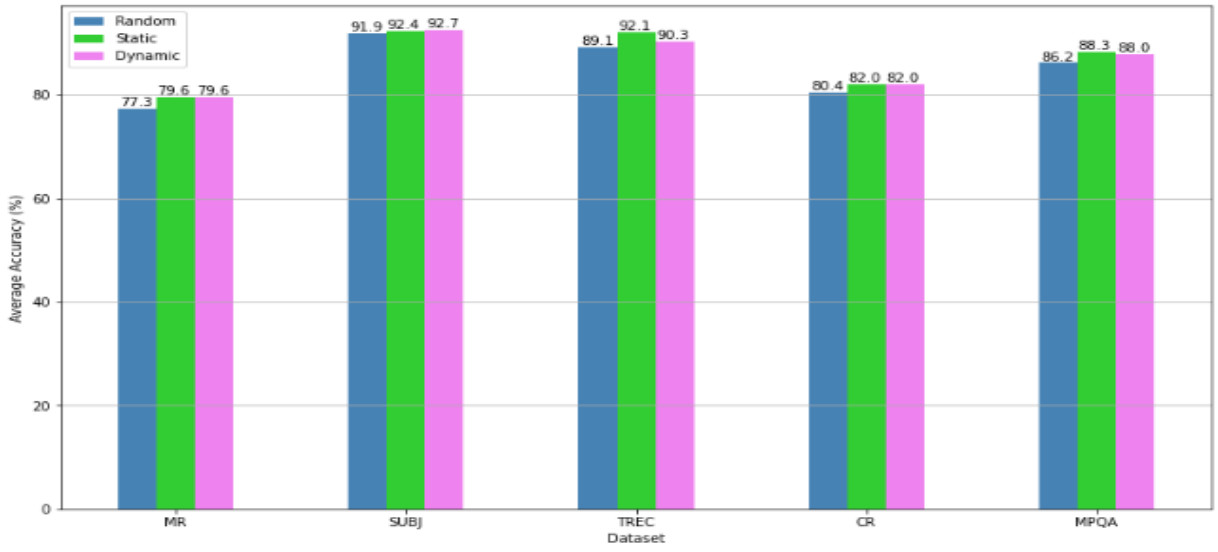
BOW Vs Word Embedding

Word embeddings and the BoW (Bag-of-Words) method are two techniques used in the processing of natural language applications, such as text categorization. The BoW approach, which is straightforward and time-tested, concentrates on the frequent use of terms in a text or document. Although it is simple to use and computationally effective, it cannot capture semantic and context-related data. on the flip hand, word embeddings are detailed vector graphics of words that include contextual and semantic information. They represent words in an eternal vector space and are trained using deep learning algorithms like Word 2 Vet, GloVe, or FastText (Kim, 2014). Word embeddings are useful for tasks like sentiment analysis, translation by machine, and question-answering that call for knowledge of the meanings of words and context. But word embedding training can be computationally costly and may need additional data. The decision between BoW

as well as word embeddings is influenced by the difficulty of the task, the data at hand, and the requirement for semantic comprehension. BoW can be a good choice for straightforward tasks using little data, while word embeddings are frequently favored for tasks requiring more nuanced knowledge of the word's meaning and context.

Random Vs Static Vs Dynamic

Fine-tuning Word embeddings may be successfully adjusted to the unique context of a task or topic by employing Word2Vec vectors with a dynamic vector representation paradigm. By updating the initial embeddings based on the data and task, the model may be able to capture task-specific subtleties and perform better. Wiebe and others, 2005b) However, fine-tuning might not always result in appreciable speed enhancements, and it might even produce less accuracy in some circumstances. The efficiency of fine-tuning can be impacted by elements including inadequate training data, overfitting, the standard of Word2Vec embeddings, task characteristics, hyperparameter tweaking, and model complexity (Kowsari et al., 2018). Evaluation of the impacts of fine-tuning on the particular task and dataset, in-depth experiments, and adequate cross-validation are all necessary to assure real gains. Additionally, when looking to enhance model performance, other strategies like feature engineering, various embeddings, or transferable knowledge from related tasks can also be taken into account.



The average accuracy between different word embedding modes.

The text classification model demonstration demonstrates that static word embedding models, such as pre-trained Word2Vec, consistently perform better than random embedding models. These

models' average accuracy is around 3% greater than that of random embeddings, showing that they effectively capture semantic information that helps them comprehend context and word connections. In text categorization tasks, this results in higher generalization and increased accuracy (Kowsari et al., 2018).

TCN Vs RNN Model

The benefits of TCNs over RNNs in sequence modelling tasks are discussed by Kowsari and Wiebe. TCNs analyses the full sequence at once, which increases their computational efficiency and speeds, particularly for lengthy sequences. While TCNs are unaffected by disappearing or bursting gradients, RNNs are. While RNNs keep a concealed state for previous information but are restricted for lengthy dependencies, TCNs are excellent for jobs involving long sequences and capturing long-range dependencies.

The best performances of Models

Top Six Deep Learning Models based on:	
the Average Accuracy Margin	the Average Rank Values
TCN-static	**Ensemble CNN-GRU-static**
Ensemble CNN-GRU-static	**TCN-static**
TCN-dynamic	**TCN-dynamic**
BiGRU-static	*1D CNN-static*
1D CNN-static	*BiGRU-static*
1D CNN-dynamic	*BiLSTM-static*

The top six deep learning models in this project.

Through testing, the top six text categorization models were determined, with the static TCN and Ensemble models achieving the best results. In terms of its an average precision margin, Static TCN, which makes use of pre-trained Word2Vec embeddings, performs better than other models. Ensemble models, which incorporate the results of numerous models, performed better at classifying texts. RNNs, 1D CNNs, and GRU, among other top-performing models (Kowsari et

al., 2018), also did well in the trials. The optimal model to use will depend on the task at hand and the resources at hand, but the fixed versions of TCN and ensemble approaches are probably your best bets for categorizing text data with high accuracy and resilience.

Conclusion

The state-of-the-art in processing natural languages problems has considerably progressed thanks to text categorization using deep learning models. By utilizing word embeds like Word2Vec, GloVe, and FastText to capture semantics and contextual information, these models have revolutionized the learning of text representations. Text classification has advanced thanks to the use of sequential learning, parallel processing, ensemble learning, transferred learning, tuning of hyper parameters, metrics for assessing model's interpretability, data pretreatment, and domain adaptability (Kim, 2014). Model accuracy is substantially impacted by properly tweaking learning rates, batch sizes, dropout rates, and model topologies. Continuous research and innovation in model architectures, word embeddings, and transfer learning continue to drive improvements in the field, making deep learning a key player in solving real-world text classification challenges. Text classification models can significantly improve accuracy and performance by using pre-trained word embeddings like Word2Vec, GloVe, and FastText. TCN, a dilated-causal version of CNN, offers a powerful alternative to recurrent architectures like RNNs. Its parallel processing capabilities enable efficient capture of long-range dependencies, making it well-suited for text classification tasks. Leveraging word embeddings and exploring innovative model architectures is crucial for achieving state-of-the-art performance in text classification using deep learning techniques.

Reference

Agarwal, B., & Mittal, N. (2014). Text Classification using Machine Learning Methods-A survey. In *Advances in intelligent systems and computing* (pp. 701–709). https://doi.org/10.1007/978-81-322-1602-5_75

Cer, D. (2018, March 29). *Universal Sentence Encoder*. arXiv.org. https://arxiv.org/abs/1803.11175

Chen, H., Wu, L., Chen, J., Lu, W., & Ding, J. (2022). A comparative study of automated legal text classification using random forests and deep learning. *Text Classification*, *59*(2), 102798. https://doi.org/10.1016/j.ipm.2021.102798

Elnagar, A., Al-Debsi, R., & Einea, O. (2020). Arabic text classification using deep learning models. *Information Processing and Management*, *57*(1), 102121. https://doi.org/10.1016/j.ipm.2019.102121

HDLTEX: Hierarchical Deep Learning for text Classification. (2017, December 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8260658/?casa_token=bt_sOHvyXEUAAAAA:xRuBae4JwlHkJY3jsmmk9Q4QvFQKWXZ5_-FIEdAu3VHUnPfCODAo3kx-KgGkke4lWcNhL_3ypg

Hu, M., & Liu, B. (2004). *Mining and summarizing customer reviews*. https://doi.org/10.1145/1014052.1014073

Kadhim, A. I. (2019). Survey on supervised machine learning techniques for automatic text classification. *Artificial Intelligence Review*, *52*(1), 273–292. https://doi.org/10.1007/s10462-018-09677-1

Kim, Y. (2014, August 25). *Convolutional neural networks for sentence classification*. arXiv.org. https://arxiv.org/abs/1408.5882

Kowsari, K., Heidarysafa, M., Brown, D. E., Meimandi, K. J., & Barnes, L. E. (2018). *RMDL*. https://doi.org/10.1145/3206098.3206111

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning--based text classification. *ACM Computing Surveys*, *54*(3), 1–40. https://doi.org/10.1145/3439726

Pang, B. (2004, September 29). *A Sentimental Education: sentiment analysis using subjectivity summarization based on minimum cuts*. arXiv.org. https://arxiv.org/abs/cs/0409058

Pang, B. (2005, June 17). *Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales*. arXiv.org. https://arxiv.org/abs/cs/0506075

Wiebe, J., Wilson, T., & Cardie, C. (2005a). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, *39*(2–3), 165–210. https://doi.org/10.1007/s10579-005-7880-9

Wiebe, J., Wilson, T., & Cardie, C. (2005b). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, *39*(2–3), 165–210. https://doi.org/10.1007/s10579-005-7880-9