The object of this study is the semantic similarity between two texts. This research focuses on developing a hybrid architecture that combines Siamese Neural Network (SNN) with Feedforward Neural Network (FNN) to measure the semantic text similarity, with text representation using Sentence-BERT (SBERT). The problem addressed is the challenge of capturing deep semantic relationships between two texts, which traditional methods, such as Term Frequency-Inverse Document Frequency (TF-IDF) or Word2Vec, find difficult to achieve. This research aims to overcome these weaknesses by combining the two architectures into a more powerful hybrid system. The test results show the highest accuracy of 87.82% on the Semantic Textual Similarity (STS) dataset using the SBERT "all-MiniLM-L6-v2" model, 76.72% on the Quora Question Pairs (QQP) dataset using the "multi-qa-MiniLM-L6-cos-v1" model, and 73.79 % on the Microsoft Research Paraphrase Corpus (MSRP) dataset using the "paraphrase-MiniLM-L12-v2" model. The optimal parameters for the number of epochs ranged from 300 to 700, and the optimal learning rate ranged from 0.01 to 0.5. SBERT models, such as "paraphrase-MiniLM-L6-v2" and "paraphrase-MiniLM-L12-v2", gave the best results on the relevant datasets. The flexibility of the "multi-qa-MiniLM-L6-cos-v1" model also shows that the model designed for question and answer tasks can be used in the paraphrase detection domain. A unique feature of the model is the integration of SBERT as a text representation, which results in a richer semantic vector than traditional methods. The model has potential for wide application in various domains, such as plagiarism detection, legal documents, and question-and-answer systems. However, implementation requires attention to parameter selection, such as learning rate and number of epochs, to avoid overfitting or underfitting

Keywords: feedforward neural network, semantic text similarity, Sentence-BERT, Siamese neural network

UDC 004.8
DOI: 10.15587/1729-4061.2025.326956

DEVELOPMENT OF A HYBRID SIAMESE AND FEEDFORWARD NEURAL NETWORKS ARCHITECTURE FOR SEMANTIC TEXT SIMILARITY MEASUREMENT

Ng Poi Wong
Doctoral Student of Computer Science,
Lecturer of Computer Science
Department of Computer Science
Universitas Sumatera Utara
Dr. T. Mansur str., 9, Medan, Indonesia, 20155
Department of Computer Science
Universitas Mikroskil
Thamrin str., 112, Medan, Indonesia, 20212
Tengku Henny Febriana Harumy
Corresponding author
Doctor of Computer Science
Department of Computer Science
Universitas Sumatera Utara
Dr. T. Mansur str., 9, Medan, Indonesia, 20155
E-mail: hennyharumy@usu.ac.id
Syahril Efendi
Doctor of Mathematics, Professor
Department of Computer Science
Universitas Sumatera Utara
Dr. T. Mansur str., 9, Medan, Indonesia, 20155

Received 27.02.2025
Received in revised form 16.04.2025
Accepted date 02.05.2025
Published date 30.06.2025

How to Cite: *Wong, N. P., Harumy, T. H. F., Efendi, S. (2025). Development of a hybrid siamese and feedforward neural networks architecture for semantic text similarity measurement. Eastern-European Journal of Enterprise Technologies, 3 (2 (135)), 30–41. https://doi.org/10.15587/1729-4061.2025.326956*

1. Introduction

In the rapidly evolving field of Natural Language Processing (NLP), the ability to measure semantic similarity between texts has become a cornerstone for various applications, ranging from information retrieval and plagiarism detection to question-answering systems and recommendation engines [1]. Semantic text similarity refers to the process of determining how closely two pieces of text convey the same meaning, even if they are expressed using different words or structures [2]. This topic is not only central to advancing NLP but also plays a critical role in addressing practical challenges across multiple domains, including healthcare [3], legal systems [4], and education [5].

The importance of semantic text similarity research lies in its potential to bridge gaps in human-computer interaction and improve the accuracy of automated systems. For instance, identifying semantically similar medical documents in healthcare can aid clinicians in making informed decisions by providing relevant case studies or treatment guidelines [3]. In the legal context, detecting similarities between legal cases ensures consistency in judicial decisions and reduces the time required for legal proceedings [4]. In education, identifying duplicate questions or plagiarism in student submissions enhances the quality of feedback and fosters academic integrity [5]. These examples underscore the critical need for robust methods to measure semantic similarity effectively.

Despite significant advancements in deep learning and neural network architectures, challenges remain in achieving high accuracy and efficiency in semantic similarity tasks. Traditional approaches, such as Term Frequency-Inverse

Document Frequency (TF-IDF) and Bag-of-Words (BoW), have been widely used but often fail to capture deeper semantic relationships due to their reliance on lexical features [6, 7]. More recent techniques, such as word embeddings (e.g., Word2Vec, GloVe) and transformer-based models like Bidirectional Encoder Representations from Transformers (BERT), have shown the capability of capturing contextual and semantic information. However, these methods still face limitations in scalability, computational efficiency, and generalizability across diverse datasets [8, 9].

Hybrid artificial neural network architectures, such as Siamese Neural Networks (SNN) combined with Feedforward Neural Networks (FNN), offer a promising direction for addressing these challenges. SNNs are particularly effective in tasks requiring pairwise comparisons, such as detecting duplicate questions or measuring document similarity [10, 11]. By integrating SNNs with FNNs, the strengths of both architectures can be harnessed: the ability of SNNs to learn shared representations and the capacity of FNNs to model complex non-linear relationships [12, 13]. Such hybrid models have the potential to enhance the precision and interpretability of semantic similarity measurements, making them more applicable to real-world scenarios.

Given the rapid pace of technological advancement, it is crucial to continuously refine and expand upon existing methodologies to keep pace with emerging demands. For example, the integration of metaheuristic approaches, such as the Protis algorithm, into neural network design has demonstrated the potential to optimize model performance and reduce uncertainty [14]. Additionally, the development of domain-specific adaptations, such as attention mechanisms and multi-task learning frameworks, further highlights the need for innovative solutions tailored to specific applications [15, 16].

Therefore, research on the development of advanced hybrid neural network architectures for semantic text similarity measurement remains highly relevant.

2. Literature review and problem statement

In the study [17], a Boolean Logic Algebra Similarity Measure (BLAB-SM) model is proposed to measure text similarity with Boolean operators. The results show that the model can effectively measure text similarity with an accuracy of 85%. However, there are limitations in measuring the similarity of long texts. The representation of Boolean logic tends to be very large and complex for long texts, making computation slow and cumbersome. In addition, Boolean logic only works at the syntactic level, making it difficult to capture deep semantic relationships between words in the text. To overcome these limitations, it is possible to use architectures that can capture semantic relationships, such as pre-trained models. Text similarity measurement utilizing Boolean algebra has certain advantages, but the challenges of handling long texts and semantic context need to be overcome to extend its application to a wider domain.

In the study [1], a Word Embedding for Semantic Similarity (WESS) model is proposed to measure the semantic similarity of short texts, with text representation using Word2Vec, GloVe, and BERT. In the pre-processing, tokenization, lowercasing, removal of special characters, and padding truncation are performed. The results show that the model achieves an accuracy of 67.5%, where word embedding containing semantic information is more effective in measuring the similarity of short texts than lexical matching-based methods. However, the accuracy obtained is still considered quite far from perfect, which is influenced by the pre-processing process performed. To overcome this weakness, text normalization and lemmatization can be added in the pre-processing process.

In study [11], a Siamese BERT model with attention mechanism is proposed to measure semantic text similarity. The results show that the model can provide a richer and more relevant semantic representation with an accuracy of 85.15% with the Quora dataset and 77.77% with the MSRP dataset. However, the attention mechanism has limitations in understanding sentences with different structures but the same meaning, which is because the attention mechanism prioritizes words that are explicitly relevant in the local context, without considering the global semantic relationship between all sentences. While in study [16], a CNN-LSTM hybrid model was proposed to identify semantically duplicate questions in the Stack Overflow community, with text representation using Word2Vec and GloVe. The results show that the model provides an accuracy of 87.09%, where the model can capture semantic relationships between texts through the generated text representations, especially by Word2Vec. However, the model has limitations in handling highly dynamic natural language variations, especially in the context of short questions containing slang or non-standard terms. Similar to the study [10], a Siamese Multi-layered Long Short-Term Memory (MaLSTM) model with Embeddings from Language Models (ELMo) is proposed to detect duplicate questions on the Quora dataset. The results show that the model achieves good performance with 88.7% accuracy. In this study, ELMo can provide a richer semantic representation than static embeddings such as Word2Vec or FastText, as ELMo can capture the dynamic context of each word in the sentence. However, it tends to be less effective in handling queries with significant syntactic variations but similar meanings, because ELMo only uses bi-LSTM to model the context, not as robust as modern transformer architectures. Although these models have their advantages in measuring semantic text similarity, they have limitations in handling global context, natural language variations, or significant syntax, so to overcome these limitations, neural network-based architectures and transformer-based word embedding models specialized for text can be used.

In the study [18], a hybrid model combining FNN with Differential Evolution and Adam was proposed to classify data with varying dimensions. The results showed that the model achieved better performance with an increase in accuracy of up to 5.85%. Similar to the study [19], which evaluated the performance of a hybrid model of metaheuristic algorithm with FNN for classification tasks in modelling and solving non-linear optimization problems. The results show that the performance of the resulting classification depends on the design of the FNN, such as the number of hidden layers, the number of neurons, and the activation function. In addition, the study [13] evaluated the performance of weight initialization methods in FNN. Results show that Xavier/Glorot initialization is superior on datasets with numeric or continuous variables. However, this approach has not been thoroughly explored in the context of other hybrid models.

To address this gap, the development of a hybrid architecture that combines SNN with FNN in semantic text similarity measurement is an appropriate solution. The resulting model

will be able to capture semantic relationships in more depth through text representation with SBERT. In addition, FNN will be used to model the non-linear relationship between text features. This is considered based on the need to overcome the limitations of traditional models that struggle to capture deep semantic relationships or have limitations in handling syntactic and text structure variations. This solution is expected to make significant contributions in practical applications such as plagiarism detection, question-answer systems, and semantic-based document classification.

3. The aim and objectives of the study

The aim of the study is to identifying text similarity by capturing semantic relationships between texts. This hybrid model aims to improve the accuracy and efficiency of text similarity measurement across various application domains, including information retrieval, plagiarism detection, question-answering systems, and document classification.

To achieve this aim, the following objectives are accomplished:

– to select models for textual representation and integrate into a hybrid architecture combining SNN with FNN to capture non-linear relationships between extracted features for semantic similarity measurement;

– to evaluate the performance of the hybrid model by assessing its effectiveness using datasets through investigating the impact of key parameters, which include learning rate, number of epochs, and embedding models, on the model's performance.

4. Materials and methods

The object of this study is the semantic similarity between two texts. The hypothesis in this study is that the integration of textual representation into a hybrid architecture that combines SNN with FNN, can measure semantic similarity between texts, due to the ability of textual representation to capture deeper semantic meaning, the ability of SNN to compare text pairs consistently, and the ability of FNN to capture non-linear relationships between text features. The underlying assumptions of this study are that text representation using a pre-trained Sentence-BERT (SBERT) model specialized for text is able to capture deeper semantic context, and the implementation of a hybrid model of SNN with FNN is able to handle variations in syntactic structure and differences in word expression in texts with similar meanings. In addition, there are several simplifications applied in this study, the datasets used are three datasets, pre-processing consists of tokenization, lowercasing, special character removal, and padding truncation, and the SBERT model used is five model variants.

The datasets used in this study are datasets taken from several open datasets, Microsoft Research Paraphrase Corpus (MSRP) taken from Microsoft Research, Semantic Textual Similarity (STS) taken from GitHub, and Quora Question Pairs (QQP) taken from Kaggle. These datasets are frequently used in NLP research, especially for tasks that involve analyzing semantic similarity or paraphrasing between text pairs. A brief description and basic information about the datasets used in this study are as follows:

1. Microsoft Research Paraphrase Corpus (MSRP) is a dataset designed to evaluate NLP models in identifying whether two sentences have the same meaning even though their word or phrase structures are different. This dataset contains sentence pairs from various sources, such as online news articles. Each pair of sentences is given a binary label with label 1 stating that the two sentences convey the same information or meaning, and label 0 stating that the two sentences have different meanings. This dataset consists of 5801 instances, which are divided into data for training, as many as 4076 instances, and data for testing, as many as 1725 instances.

2. Semantic Textual Similarity (STS) is a dataset introduced in the annual Semantic Evaluation (SemEval) competition, sponsored by the Association for Computational Linguistics (ACL), and aims to measure the degree of semantic similarity between two sentences. The STS dataset provides a continuous numerical score by calculating how similar two sentences are in terms of meaning, which indicates the degree of semantic similarity. This dataset contains sentence pairs from various types of texts, such as dialogues, news headlines, and others. Each pair of sentences is assigned a numerical label value in the range of 0 to 5, with a value of 5 stating that the two sentences have the same semantic relationship, while a value of 0 states that the two sentences have no semantic relationship at all. This dataset consists of 15250 instances, which are divided into data for training, as many as 3,000 instances, and data for testing, as many as 12250 instances.

3. Quora Question Pairs (QQP) is a dataset released by the question-and-answer platform Quora that is specifically designed to detect question duplication. This dataset contains pairs of questions taken from the Quora platform. Each pair of questions has a binary label with label 1 stating that the two questions have the same meaning, and label 0 stating that the two questions have different meanings. This dataset consists of 1452876 instances, which are divided into data for training, as many as 404301 instances, and data for testing, as many as 1048575 instances.

In this study, all instances of the MSRP and STS datasets will be used in training and testing to evaluate their performance in text similarity measurement. Before the STS dataset is used, all numeric label values in the range of 0 to 5 will be converted into binary labels, as shown below:

$$\text{Binary label} = \begin{cases} 1, \text{numeric label} > 3, \\ 0, \text{numeric label} \le 3. \end{cases}$$

As for the QQP dataset, considering computational resources, only a small number of instances from all existing QQP datasets were used, 4983 instances were taken as training data, and 2492 instances as testing data. All datasets consist of two parts: training data, which is used to train the model, and testing data, which is used to test the performance of the model after training. To monitor the performance of the model during training, the training data will be divided into training data and validation data with a ratio of 80:20.

In NLP, especially in text similarity measurement, pre-processing is a crucial first step to transform raw text into a format that can be further processed by machine learning models. The purpose of pre-processing is to clean and normalise the text to improve the quality of input for machine learning models. By removing irrelevant information and simplifying the text structure, the model can focus more on features that are meaningful in determining similarities be-

tween texts. Pre-processing involves a series of techniques to clean and normalise text data. In this study, the pre-processing steps performed include:

1. Data cleaning to remove invalid or redundant data. Data cleaning consists of removing data with empty values in one of the text pairs, removing data with identical text pairs to avoid bias in model learning, and ensuring the data type format uniformity of all values in the text column is a string data type.

2. Text normalization to ensure that the text to be used in the model has a uniform format. Text normalization consists of converting to lowercase to avoid differences in meaning due to capitalization, removing non-alphanumeric characters such as symbols, punctuation marks, and special characters to retain only letters and numbers, and removing excessive space characters to ensure cleaner and more consistent text.

3. Tokenization, which breaks the text into tokens or word units using the spaCy NLP library.

4. Removal of conjunctions, where the words do not contribute significantly to the meaning of the text, to reduce noise in the words.

5. Lemmatization, which converts words into their basic forms so that variations of words that have similar meanings can be put together.

Artificial Neural Network (ANN) is a computational model inspired by the way the human brain works in processing information through a network of neurons, which can work with multidimensional data, handle non-linear relationships, and produce predictions or classifications with a high level of accuracy [20].

Text representation is an important step that converts raw text data into a numerical format that can be understood by machine learning models. Text representation methods have evolved from simple approaches such as BoW and TF-IDF towards more sophisticated techniques such as word embeddings [6, 21]. However, these methods have limitations in capturing the semantic meaning and context of the words in the text. To overcome this problem, word embeddings such as Word2Vec and GloVe were developed to represent words in the form of low-dimensional vectors by considering their context, but these models still lack in capturing the deeper contextual meaning of a sentence or document [2, 8]. Transformers-based models such as Bidirectional Encoder Representations from Transformers (BERT) were developed to generate text representations that consider the context from both directions, making them more effective in understanding the semantic meaning of text, but these models were designed for tasks such as classification or question answering, and are less efficient for generating directly comparable sentence embeddings [9]. Therefore, Sentence-BERT (SBERT) was developed as a modification of BERT that uses Siamese and triplet network architectures to generate semantic sentence representations [8].

In this study, text representation uses five SBERT models to convert text pairs into numerical vectors: 'all-MiniLM-L6-v2', 'all-MiniLM-L12-v2', 'paraphrase-MiniLM-L6-v2', 'paraphrase-MiniLM-L12-v2', and 'multi-qa-MiniLM-L6-cos-v1' models. According to www.sbert.net, the 'all-MiniLM-L6-v2' model is designed for computational efficiency without significantly compromising the quality of text representation and is suitable for text classification, clustering, and semantic similarity measurement tasks between short texts. The 'all-MiniLM-L12-v2' model is a more complex version of the 'all-MiniLM-L6-v2' model and is effective in capturing deeper semantic relationships, especially in long or complex texts. The 'paraphrase-MiniLM-L6-v2' model produces excellent text representations for comparing pairs of sentences with similar meanings but different word expressions and is widely used in duplicate document filtering and semantic similarity measurement applications. The 'paraphrase-MiniLM-L12-v2' model is a more complex version of the 'paraphrase-MiniLM-L6-v2' model that excels in capturing deeper semantic relationships, especially in long texts and is suitable for applications that require high accuracy in text similarity measurement. The 'multi-qa-MiniLM-L6-cos-v1' model is optimised for question-answer (QA) tasks, produces an embedding suitable for cosine similarity calculation, and is effective in measuring semantic similarity between queries and documents. Although the 'multi-qa-MiniLM-L6-cos-v1' model is not specifically designed for paraphrase detection, it shows good flexibility in other applications such as text classification. The five models produced a numerical vector of dimension 384. The selection of these models is based on computational resource efficiency considerations to evaluate performance in text similarity measurement tasks.

FeedForward Neural Network (FNN) is one of the basic architectures of ANN, where information flows in one direction, starting from the input layer, through one or more hidden layers, until it reaches the output layer without any feedback loops [22], as shown in Fig. 1. FNN was first introduced based on the perceptron model and became the basis for many developments in deep learning [13, 23]. In this study, FNN is used as an identical sub-network within SNN.

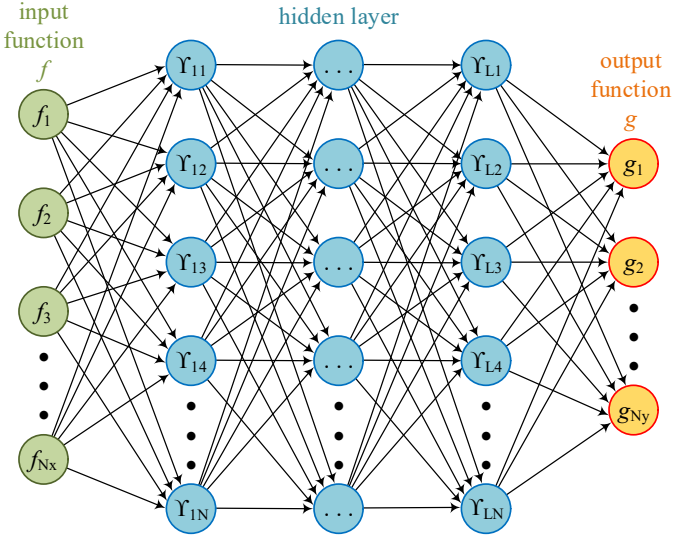

Fig. 1. Feedforward neural network [22]

Siamese Neural Network (SNN) is an artificial neural network architecture consisting of two or more identical subnetworks that share weights and are used to compare two inputs, as shown in Fig. 2. SNN was first introduced for tasks such as face recognition and fingerprint verification, but later evolved into a highly effective tool in NLP for tasks such as text similarity detection, question matching, and semantic analysis [8, 12]. The main advantage of SNN lies in its ability to compare pairs of inputs efficiently, using the vector representation generated by the identical subnetwork to calculate the degree of similarity between two inputs [3, 24].

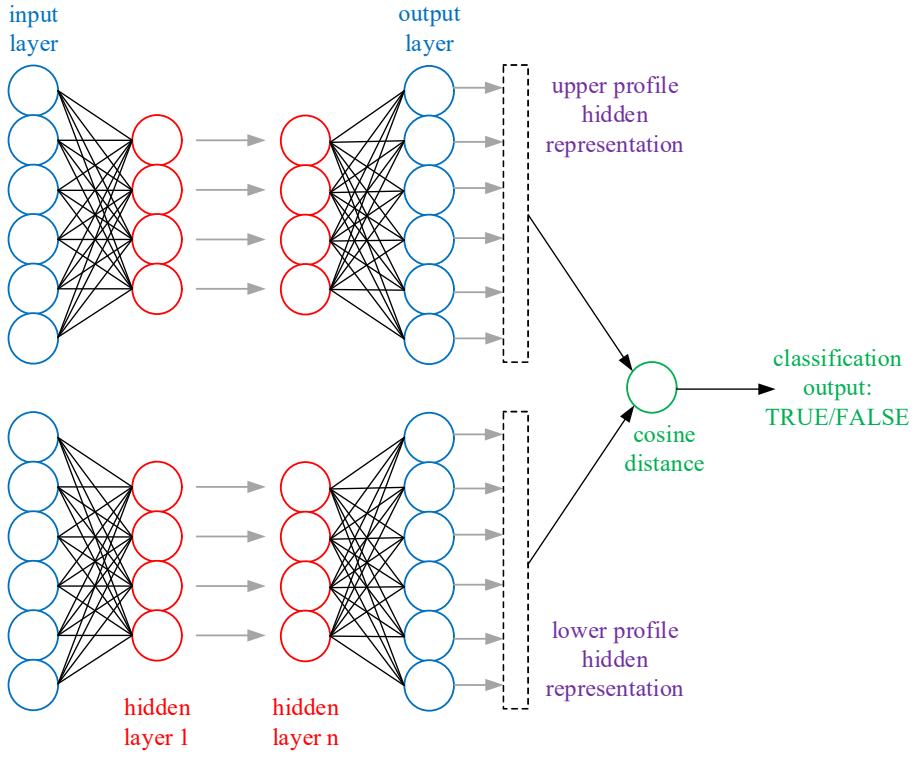

Fig. 2. Siamese neural network [12]

In this study, SNN is used as the main component in a hybrid architecture for similarity measurement between two texts. The identical network in SNN consists of two identical sub-networks with FNN architecture, where both identical sub-networks will share weights and biases. The output of the two sub-networks will be compared using Cosine Similarity to produce a predicted text similarity value. The results of the prediction value will then be calculated loss function using Binary Cross-Entropy (BCE) to measure the extent of the difference between the predicted value and the actual value of text similarity. The results of the prediction value and loss function will then be used in backpropagation with Gradient Descent to update the weight and bias values, where the new weight and bias values will be used in the next iteration in the identical sub-network.

Evaluation is an important step in the process of developing a machine learning system to measure the performance of the model against the dataset used. The evaluation metric used in this study is accuracy, which gives an idea of how accurate the model is in predicting the actual label by comparing it to the model prediction.

Accuracy is calculated based on the components of the confusion matrix, consisting of four main components: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), as can be seen in Table 1. To calculate the accuracy value, the following formula is used below

$$\text{Accuracy}=\frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}. \quad (2)$$

Table 1

Confusion matrix

Text similarity measurement		Prediction label	
		1 (Similar)	0 (Dissimilar)
Actual label	1 (Similar)	True Positive (TP)	False Negative (FN)
	0 (Dissimilar)	False Positive (FP)	True Negative (TN)

The development of a hybrid architecture that combines SNN with FNN, and its performance evaluation is carried out with the assistance of Google Collaboratory or also known as Google Colab, which is a cloud-based platform with Jupyter Notebook format for writing, running, and sharing Python code through a web browser. This platform can be used to access programming environments for free, supporting various purposes such as data science, machine learning, and deep learning. In addition, with the collaboration feature, Google Colab allows researchers and programmers to work collaboratively on programming projects without having to install additional software on their local computers. In using Google Colab with Python, it utilises CPU RAM of 12.7 GB and Disk of 107.7 GB. In developing the hybrid architecture of SNN and FNN with Python, the Pandas library for dataset processing, SpaCy and nltk, which are NLP libraries for pre-processing, sentence_transformers to generate text embedding using pre-trained Sentence-BERT models, Tensorflow, which is a machine learning framework for building, training, and evaluating neural network models, and Sklearn, which is a machine learning library that provides tools for modelling, evaluation, and pre-processing.

5. Results of the hybrid architecture combining SNN with FNN for semantic text similarity measurement

5. 1. Integration of textual representation into a hybrid architecture combining SNN with FNN for semantic similarity measurement

In integrating textual representation into a hybrid architecture that combines SNN with FNN for semantic text similarity measurement, the first step after pre-processing is text representation. Text representation in this study uses SBERT to convert text pairs into numerical vectors. There are five models of SBERT used in text representation: 'all-MiniLM-L6-v2', 'all-MiniLM-L12-v2', 'paraphrase-MiniLM-L6-v2', 'paraphrase-MiniLM-L12-v2', and 'multi-qa-MiniLM-L6-cos-v1' models. According to www.sbert.net, the 'all-MiniLM-L6-v2' model is designed for computational efficiency without significantly compromising the quality of text representation and is suitable for text classification, clustering, and semantic similarity measurement tasks between short texts. The 'all-MiniLM-L12-v2' model is a more complex version of the 'all-MiniLM-L6-v2' model and is effective in capturing deeper semantic relationships, especially in long or complex texts. The 'paraphrase-MiniLM-L6-v2' model produces excellent text representations for comparing pairs of sentences with similar meanings but different word expressions and is widely used in duplicate document filtering and semantic similarity measurement applications. The

'paraphrase-MiniLM-L12-v2' model is a more complex version of the 'paraphrase-MiniLM-L6-v2' model that excels in capturing deeper semantic relationships, especially in long texts and is suitable for applications that require high accuracy in text similarity measurement. The 'multi-qa-MiniLM-L6-cos-v1' model is optimised for question-answer (QA) tasks, produces an embedding suitable for cosine similarity calculation, and is effective in measuring semantic similarity between queries and documents. Although the 'multi-qa-MiniLM-L6-cos-v1' model is not specifically designed for paraphrase detection, it shows good flexibility in other applications such as text classification. The five models produced a numerical vector of dimension 384. The selection of these models is based on computational resource efficiency considerations to evaluate performance in text similarity measurement tasks.

FNN in this study is used as an identical sub-network within SNN. The FNN process starts by receiving input in the form of numeric vectors generated by SBERT. The numeric vector is then processed through a hidden layer consisting of several neurons. The neurons are calculated in a linear combination, and then a non-linear transformation using the Tanh activation function. The output of the hidden layer then proceeds to the output layer, a linear combination is calculated, and a non-linear transformation is performed using a Sigmoid activation function to produce a probability of similarity between two texts. The output of the output layer that comes from two identical sub-networks will then be compared using Cosine Similarity to produce a final prediction in the form of a text similarity value.

SNN in this study is used as the main component in a hybrid architecture for similarity measurement between two texts. The SNN process starts by receiving two input texts, which are then subjected to pre-processing and text representation. The numerical vector representations of the two texts are input into two identical sub-networks with FNN architecture, where the two identical sub-networks will share weights and biases. The first sub-network receives input in the form of the first numeric vector from the first text, and the second sub-network receives input in the form of the second numeric vector from the second text. The output of the two identical sub-networks will be compared using Cosine Similarity to produce a predicted value of text similarity. The results of this comparison will then be used to calculate the loss function with Binary Cross-Entropy (BCE), which serves to measure the extent of the difference between the predicted value of the model and the actual text similarity value. The results of the loss function will be the basis for the backpropagation process using Gradient Descent to update the new weight and bias values, where the new weight and bias values will be reused in the next iteration.

The result of developing a hybrid architecture that combines SNN and FNN to measure the semantic similarity between two texts overcomes the limitations of traditional models by utilising the advantages of both architectures. As shown in Fig. 3, the steps resulting from the development of a hybrid architecture combining SNN with FNN for semantic text similarity measurement are as follows:

1. Take two sentences from the dataset used, Text 1 and Text 2.

2. Perform pre-processing on both texts, which consists of cleaning empty data, cleaning duplicate data, text normalization, tokenization, conjunction removal, and lemmatization.

3. Perform the text representation process by converting both texts from the pre-processing results into numeric vectors using Sentence-BERT (SBERT).

4. The numeric vectors from the text representation results will be input into the input layer of the SNN, which in this SNN will consist of two identical sub-networks. Vector 1 will be input into the input layer of the first sub-network, and vector 2 will be input into the input layer of the second sub-network.

5. Generate the initial values of weights and bias using Xavier/Glorot Initialization.

6. Each identical sub-network, will consist of several hidden layers and output layers. Each hidden layer and output layer will use shared weights and biases. In the hidden layer, calculate the linear combination and Tanh activation, while in the output layer, calculate the linear combination and sigmoid activation.

7. Check the similarity of the two outputs of the output layer of the two identical sub-networks, using cosine similarity.

8. Calculate the loss function based on the similarity value of the actual text with the result of the similarity metric using Binary Cross-Entropy (BCE).

9. Perform the backpropagation process using gradient descent to generate new weights and biases based on the loss function results, where the new weights and biases will be used in the hidden layer and output layer in the next iteration.

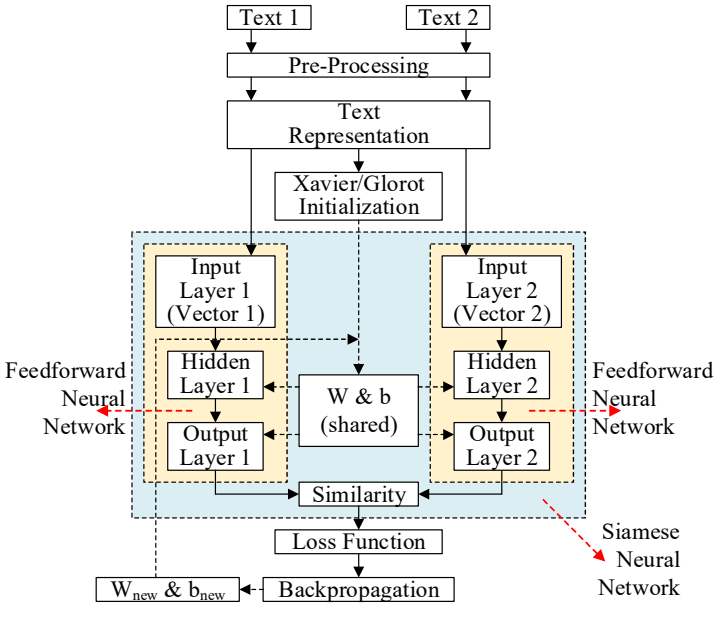

Fig. 3. A hybrid architecture of SNN with FNN combination

5. 2. Evaluation of hybrid model performance in measuring semantic text similarity

A hybrid architecture model combining SNN and FNN was designed and then implemented in Python. The model is then evaluated for performance using the Accuracy on three provided datasets (MSRP, STS, and QQP) with several testing parameters such as embedding model, number of epochs, and learning rate. Text representation into numeric vectors uses five variants of SBERT pre-trained embedding models, where all models produce numeric vectors of dimension 384. For the number of epochs parameter, the number of epochs ranging from 100 to 1000 with a multiple of 100 is used, while for the learning rate parameter, 0.5, 0.1, 0.05, 0.01, 0.005, and 0.001 are used. For optimization in updating the model weights during training, the Adam Optimizer was used.

As can be seen in Tables 2–6, these tables present the test results in the form of accuracy in percentage units for each SBERT model used in the test. Table 2 presents test results with the 'all-MiniLM-L6-v2' model, Table 3 presents test results with the 'all-MiniLM-L12-v2' model, Table 4 presents test results with the 'paraphrase-MiniLM-L6-v2' model, Table 5 presents test results with the 'paraphrase-MiniLM-L12-v2' model, and Table 6 presents test results with the 'multi-qa-MiniLM-L6-cos-v1' model. From these tables, the row states the learning rate used in the test for each dataset (MSRP, STS, and QQP), while the column states the number of epochs used in the test.

As can be seen in Table 7, this table presents a recapitulation of the test results in the form of the best accuracy in percentage units based on the test results from Tables 2–6. In the table, the row states the SBERT model used in the test, while the column states the best accuracy value, the number of epochs that produce the best accuracy, and the learning rate used in producing the best accuracy. Each accuracy value, number of epochs, and learning rate is presented for each dataset used (MSRP, STS, and QQP).

Based on Table 7, when viewed from the SBERT model used has a significant influence on model performance. The 'all-MiniLM-L6-v2', 'paraphrase-MiniLM-L12-v2' and 'multi-qa-MiniLM-L6-cos-v1' models provide better accuracy than other models, with the following explanation:

Table 2

Accuracy results with the 'all-MiniLM-L6-v2' model (in percentage)

Dataset	LR	Epochs									
		100	200	300	400	500	600	700	800	900	1000
MSRP	0.5	72.39	71.1	71.75	69.06	65.56	70.93	66.02	66.26	71.45	66.84
	0.1	67.78	67.31	66.49	66.08	67.78	65.67	66.2	64.92	70.81	71.34
	0.05	65.27	59.84	61.53	60.71	61.59	61.76	61.53	62.76	63.05	59.89
	0.01	63.34	60.42	63.46	60.65	61.65	60.48	60.6	60.25	60.95	58.9
	0.005	67.43	64.33	60.71	62.05	62.41	60.71	61.59	60.71	61.18	60.83
	0.001	68.24	69.47	69.06	66.73	64.51	64.57	63.46	63.46	62.29	62.64
STS	0.5	87.35	86.54	86.82	87.82	87.45	80.77	79.41	78.42	82.59	81.18
	0.1	85.41	84.82	84.56	83.75	84.94	82.38	83.58	82.55	84.84	83.78
	0.05	84.23	85.06	84.25	84.82	84.11	83.78	84.37	83.84	83.82	83.68
	0.01	84.99	84.88	83.93	84.22	84.02	84.09	84.38	84.12	84.59	83.82
	0.005	84.23	85.2	84.22	83.73	83.75	84.26	83.71	83.56	84.18	84.69
	0.001	50.46	74.91	83.18	83.59	82.67	82.99	82.62	82.82	82.05	82.41
QQP	0.5	49.48	49	49.6	50.08	50.76	48.84	49.36	48.43	50.2	49.56
	0.1	73.03	70.3	73.27	49.84	50.4	51	49.88	50.56	50.16	49.16
	0.05	72.95	72.91	49.04	50.24	49.84	50.32	50.8	51.85	48.92	51.97
	0.01	71.91	73.23	72.23	73.31	72.47	71.51	72.19	48.76	49.48	49.64
	0.005	70.71	71.67	72.15	71.15	71.31	72.07	72.07	70.99	71.87	72.63
	0.001	40.81	59.95	64.85	67.13	68.5	68.26	68.62	69.9	70.14	68.5

Notes: LR – learning rate.

Table 3

Accuracy results with the 'all-MiniLM-L12-v2' model (in percentage)

Dataset	LR	Epochs									
		100	200	300	400	500	600	700	800	900	1000
MSRP	0.5	71.69	73.15	71.1	69.18	68.13	67.6	71.1	68.48	67.37	67.83
	0.1	65.21	64.51	64.51	65.09	63.51	64.1	65.62	64.74	65.44	70.75
	0.05	63.51	61.47	61.7	60.3	63.16	61.24	60.65	59.25	62.17	60.95
	0.01	63.92	60.36	62.46	62.05	60.48	60.89	60.71	60.3	59.43	60.83
	0.005	66.43	63.46	61.82	61.65	64.04	61.82	62.11	60.48	61.3	60.19
	0.001	67.78	68.77	68.07	65.79	63.28	62.99	63.57	63.16	63.28	63.4
STS	0.5	87.01	87.38	87.46	87.1	81.62	85.03	79.86	81.09	76.77	85.61
	0.1	84.99	85.37	84.85	84.62	84.73	84.78	84.91	85.37	83.94	85
	0.05	84.55	83.64	84.08	84.4	84.17	84.02	84.64	84.03	84.5	83.81
	0.01	85.25	84.79	84.52	84.41	83.5	84.61	84.87	84.34	84.09	83.81
	0.005	84.12	84.38	84.2	84.25	83.56	84.73	83.62	84.09	83.37	83.79
	0.001	49.34	74.19	81.96	82.61	82.7	84.25	83.93	81.77	81.55	83.23
QQP	0.5	49.16	50.96	49.48	49.4	49.68	49.32	49.4	49.56	50.36	50.12
	0.1	72.71	73.11	70.67	71.87	72.23	71.55	72.91	70.99	74.16	50.28
	0.05	70.63	71.55	51.61	49.16	51.32	52.29	49.28	50.08	49.76	51.69
	0.01	71.87	71.91	72.75	70.95	69.82	71.99	72.67	51.28	48.72	48.11
	0.005	70.26	70.67	71.19	70.83	70.26	71.79	72.03	70.79	70.22	70.67
	0.001	41.65	60.11	64.89	67.42	69.18	68.14	66.73	68.02	66.89	67.05

Notes: LR – learning rate.

Table 4

Accuracy results with the 'paraphrase-MiniLM-L6-v2' model (in percentage)

Dataset	LR	Epochs									
		100	200	300	400	500	600	700	800	900	1000
MSRP	0.5	73.73	71.75	71.8	69.29	69.82	69.35	67.43	67.54	69.06	69.47
	0.1	69.99	70.46	70.23	69.53	70.64	71.16	68.77	69	70.75	69.12
	0.05	65.85	64.97	65.79	64.04	64.16	65.09	65.91	66.43	64.57	71.86
	0.01	63.46	62.41	62.41	62.05	62.81	60.07	61.53	61.12	59.6	61.06
	0.005	65.09	63.57	62.7	63.51	62.35	60.19	62.46	61.59	62.29	62.29
	0.001	71.98	66.96	65.38	63.51	64.74	64.68	63.63	64.04	64.33	61.59
STS	0.5	85.41	85.11	82.56	80.86	78.24	74.56	81.55	79.17	79.06	78.61
	0.1	86.64	86.5	85.49	86.57	86.99	86.87	86.49	86.34	85.03	86.76
	0.05	86.11	86.34	85.96	85.31	86.66	84.35	84.55	84.26	83.99	86.11
	0.01	86.41	86.32	86.28	85.41	85.32	85.79	85.78	85.72	85.06	85.13
	0.005	85.97	85.82	85.7	85.4	85.91	85.25	84.84	85.34	85.94	85.08
	0.001	81.99	85.72	85.11	85.14	85.05	85.32	84.78	84.79	85.09	85.14
QQP	0.5	72.83	73.52	49.32	49.04	49.8	49.72	50.4	50.64	50.44	48.92
	0.1	72.95	71.63	72.51	75.36	75.48	49.84	50.88	50.12	49.16	50.24
	0.05	72.03	73.39	72.23	48.19	49.88	51.12	49.12	49.24	48.39	49.24
	0.01	74.68	73.84	73.6	72.51	71.99	73.07	72.03	72.71	47.83	49.68
	0.005	74.32	73.11	72.51	71.27	73.56	71.95	72.15	72.75	72.79	71.99
	0.001	67.3	70.26	71.11	71.19	71.11	70.35	70.02	70.71	70.47	71.55

Notes: LR – learning rate.

Table 5

Accuracy results with the 'paraphrase-MiniLM-L12-v2' model (in percentage)

Dataset	LR	Epochs									
		100	200	300	400	500	600	700	800	900	1000
MSRP	0.5	72.62	73.79	72.91	70.99	70.34	70.46	67.78	70.29	70.34	67.37
	0.1	69.88	71.34	68.13	71.51	69.99	69.64	70.05	71.8	73.61	70.69
	0.05	65.03	63.57	64.1	65.62	66.49	63.92	64.92	64.86	65.15	62.64
	0.01	64.16	63.22	63.98	61.06	62.29	62.11	62.52	64.33	61.94	61.24
	0.005	65.5	65.21	63.16	62.58	63.75	63.51	61.88	62.17	61.35	61.65
	0.001	71.63	67.43	67.08	65.32	66.26	63.86	65.03	63.86	63.28	64.21
STS	0.5	85.56	85.32	82.81	84.67	84.37	84.66	83.31	82.9	85.7	83.44
	0.1	87.17	86.76	86.9	87.07	86.16	86.85	86.5	87.16	86.19	86.26
	0.05	86.43	86.34	85.87	86.37	86.84	86.6	86.22	86.64	86.03	84.59
	0.01	86.44	85.87	85.87	85.43	84.62	85.97	85.78	85.67	85.2	85.79
	0.005	86.54	86.2	85.9	85.47	85.81	85.06	85.78	85.7	85.37	85.49
	0.001	79.97	85.09	85.11	85.14	85.46	84.73	85.43	85.09	84.78	84.82
QQP	0.5	73.15	74.24	51.28	47.79	49.36	49.08	48.92	50.76	49.76	48.6
	0.1	75.32	72.51	71.31	72.55	73.52	74.04	72.35	73.23	71.87	73.56
	0.05	75.12	73.43	71.79	72.51	48.84	49.24	51.16	49.8	48.88	50.76
	0.01	73.56	74.16	74.56	73.27	72.23	72.43	51	52.09	48.88	50.32
	0.005	72.83	73.43	74.84	72.55	71.59	71.95	74.2	71.31	72.07	72.47
	0.001	63.44	69.98	70.39	69.5	69.54	69.5	70.75	69.34	70.51	70.02

Notes: LR – learning rate.

1. With the 'all-MiniLM-L6-v2' model on the STS dataset with the number of epochs 400 and learning rate of 0.5, provides the best accuracy of 87.82%, as can be seen in Tables 2, 6. This performance is likely due to the model's ability to capture general semantic representations relevant to the STS dataset.

2. With the 'paraphrase-MiniLM-L12-v2' model on the MSRP dataset with several epochs of 200 and a learning rate of 0.5, it provides the best accuracy of 73.79%, as can be seen in Tables 5, 6. The performance results show that this model is specifically optimised for paraphrase detection tasks so that it can capture semantic relationships between text pairs well.

3. With the 'multi-qa-MiniLM-L6-cos-v1' model on the QQP dataset with several epochs of 300 and a learning rate of 0.1, provides the best accuracy of 76.72%, as can be seen in Tables 6, 7. Although this model is more optimised for question-and-answer tasks, the performance results show that it has enough flexibility to be used in the context of paraphrase detection tasks.

The choice of the SBERT model that is tailored to the characteristics of the dataset and task greatly affects the performance of the model. Models optimised for paraphrase detection tasks tend to give better results.

Table 6

Accuracy results with the 'multi-qa-MiniLM-L6-cos-v1' model (in percentage).

Dataset	LR	Epochs									
		100	200	300	400	500	600	700	800	900	1000
MSRP	0.5	69.99	70.23	70.34	68.77	69.82	68.77	68.71	66.55	66.37	70.4
	0.1	66.08	66.26	65.44	66.67	64.68	65.44	70.93	64.68	61.76	69.7
	0.05	61.18	61.06	60.77	61.06	63.4	58.49	63.86	62.05	61.35	60.48
	0.01	60.89	63.05	61.7	61.82	60.65	61.35	59.43	59.19	58.79	60.48
	0.005	65.62	62.7	62.7	59.37	60.48	61.18	59.49	58.84	58.09	60.01
	0.001	67.54	69.41	66.9	64.74	63.98	63.51	63.16	63.46	61.53	61.59
STS	0.5	86.97	87.11	78.21	86.52	78.38	72.93	76.57	79.88	79.47	65.58
	0.1	84.11	81.14	83.43	85.26	84.05	85.61	83.17	82.59	83.46	83.75
	0.05	84.14	83.87	84.15	84.94	83.97	83.03	83.62	84.15	83.97	83.49
	0.01	84.2	84.15	83.53	83.59	84.08	83.37	83.67	84.76	82.84	84.12
	0.005	84.67	84.23	83.99	83.82	83.2	82.84	82.99	83.84	82.96	83.47
	0.001	50.17	72.95	80.96	81.62	81.41	82.43	82.2	82.5	82.24	82.82
QQP	0.5	74.64	75.44	76.73	49	48.23	50.56	49.92	49.4	50.36	49.96
	0.1	72.95	50.08	50.92	49.64	50.72	51.52	51.16	50.88	49.32	49.48
	0.05	72.43	50.44	50.64	51.93	50.4	49.24	49.28	50.4	50.48	50.04
	0.01	70.75	72.27	71.51	71.95	70.95	72.27	71.35	49.92	50.52	50.32
	0.005	72.15	71.43	71.15	70.91	72.51	71.27	71.39	71.19	72.59	71.59
	0.001	40.17	60.55	64.93	67.34	68.5	67.98	68.74	67.17	68.38	70.59

Notes: LR – learning rate.

Table 7

Recapitulation of the best accuracy results (in percentage)

Model	Dataset = MSRP			Dataset = STS			Dataset = QQP		
	Acc	Epc	LR	Acc	Epc	LR	Acc	Epc	LR
all-MiniLM-L6-v2	72.39	100	0.5	87.82	400	0.5	73.31	400	0.01
all-MiniLM-L12-v2	73.15	200	0.5	87.46	300	0.5	74.16	900	0.1
paraphrase-MiniLM-L6-v2	73.73	100	0.5	86.99	500	0.1	75.48	500	0.1
paraphrase-MiniLM-L12-v2	73.79	200	0.5	87.17	100	0.1	75.32	100	0.1
multi-qa-MiniLM-L6-cos-v1	70.93	700	0.1	87.11	200	0.5	76.72	300	0.5

Notes: Acc – accuracy; Epc – epochs; LR – learning rate.

Based on Tables 2–6, when viewed in terms of the number of epochs used, it has a significant effect on model performance. The model performance tends to increase until a certain epoch, after which it starts to stagnate or even decrease due to overfitting. Some findings from the test results analysed based on the number of epochs are as follows:

1. At a low number of epochs, for example, between 100 and 200, the model has not fully converged, resulting in relatively low performance. One of them can be seen in Table 6 with the 'multi-qa-MiniLM-L6-cos-v1' model on the STS dataset and a learning rate of 0.001, the accuracy with the number of epochs 100 is only obtained at 50.17%, while with the number of epochs 200, the accuracy increases to 72.95%. However, in some cases, the model can obtain its best accuracy with a low number of epochs, as can be seen on the MSRP dataset with the number of epochs between 100 and 200 and a learning rate of 0.5 for all models, except the 'multi-qa-MiniLM-L6-cos-v1' model. This is likely due to the characteristics of the MSRP dataset, which is simpler and less complex than other datasets, so the model does not require many iterations to learn.

2. At an intermediate number of epochs, for example, between 300 and 700, the model performance can reach its best accuracy. On the MSRP dataset with the 'multi-qa-MiniLM-L6-cos-v1' model and learning rate of 0.1, the best accuracy of 70.93% is obtained with the number of epochs 700, as can be seen in Tables 6, 7. On the STS dataset with the 'all-MiniLM-L6-v2' model and learning rate of 0.5, the best accuracy is 87.82% with the number of epochs 400, the 'all-MiniLM-L12-v2' model and learning rate of 0.5, the best accuracy is 87.46% with the number of epochs 300, and the 'paraphrase-MiniLM-L6-v2' model and learning rate of 0.1, the best accuracy is 86.99% with the number of epochs 500, as can be seen in Tables 2–3, 6. While on the QQP dataset with the 'all-MiniLM-L6-v2' model and learning rate of 0.01, the best accuracy was 73.31% with the number of epochs 400, the 'paraphrase-MiniLM-L6-v2' model and learning rate of 0.1 obtained the best accuracy of 75.48% with the number of epochs 500, and the 'multi-qa-MiniLM-L6-cos-v1' model and learning rate of 0.5 obtained the best accuracy of 76.72% with the number of epochs 300, as can be seen in Tables 2, 4–6. This epoch range is often the optimal point where the model has learnt enough without overfitting. These test results show the importance of choosing the right number of epochs to achieve a balance between underfitting and overfitting.

3. At high epoch counts, for example, starting from 800 epoch counts, the performance starts to decrease due to overfitting. One of them can be seen in Table 3 with the 'all-MiniLM-L12-v2' model on the QQP dataset and learning rate of 0.01, the accuracy with the number of epochs 800 drops to 51.28% from the previous 72.67% with the number of ep-

ochs 700. However, in some cases, a high number of epochs can provide the best accuracy if the model manages to learn more in-depth features, such as with the 'all-MiniLM-L12-v2' model on the QQP dataset and learning rate of 0.1, the best accuracy is 74.16% with the number of epochs 900. This is likely due to the combination of larger and more complex datasets, as well as the ability of the model to utilise the additional epochs effectively without overfitting.

The optimal number of epochs is highly dependent on the characteristics of the dataset and other parameters such as the learning rate. For simpler datasets, a low to medium number of epochs is sufficient to achieve the best performance. However, for more complex datasets, a medium number of epochs generally gives the best performance. High epoch counts should be used with caution due to the high risk of overfitting, although in some cases, it can provide good performance if the model and dataset are favourable.

Based on Tables 2–6, when viewed from the learning rate used, it shows that the learning rate also plays an important role in the stability and convergence of the model. A learning rate that is too high tends to cause the training to be unstable, while a learning rate that is too low causes slow convergence. Some findings from the test results analysed based on the learning rate are as follows:

1. The optimal learning rate that produces the best accuracy is generally in the range of 0.1 to 0.5, as can be seen in Table 7. In certain cases, such as in the QQP dataset, the optimal learning rate is 0.01 with the 'all-MiniLM-L6-v2' model and the number of epochs 400, the best accuracy is 73.31%. This is likely due to the more complex characteristics of the QQP dataset, so a lower learning rate allows the model to learn slowly but stably.

2. A learning rate between 0.5 to 0.01 can give less stable results with the QQP dataset. One of them can be seen in Table 2 with the 'all-MiniLM-L6-v2' model, where with a learning rate of 0.5 the accuracy is between 49% to 50%, with a learning rate of 0.1 the accuracy is above 70% with the number of epochs 100 to 300 and will drastically decrease to the range between 49% and 50 % starting from the number of epochs 400, to with a learning rate of 0.01 the accuracy is above 70% with the number of epochs 100 to 700 and will drastically decrease to <50% starting from the number of epochs 800. This is likely due to the weight update step being too large so that the model fails to converge to a local or global minimum.

3. Learning rates that are too low, such as 0.001, tend to cause slow convergence and suboptimal results. This can be seen in the QQP dataset in Table 2 to Table 6, where the resulting accuracy tends to be in the range between 68% and 70% compared to the learning rate of 0.005, which is in the range between 70% and 75%. This is likely due to the weight update step being too small, and the model needs more iterations to learn the relevant patterns.

The selection of an appropriate learning rate is highly dependent on the characteristics of the dataset and the SBERT model used. The optimal learning rate is generally in the range of 0.01 to 0.5, which is adjusted to the dataset and the number of epochs.

6. Discussion of results of the development of a hybrid architecture combining SNN with FNN for semantic text similarity measurement

This study shows that the integration of textual representation into a hybrid architecture combining SNN with FNN as shown in Fig. 3, can measure the semantic similarity of texts on the dataset used. The focus on the numerical vector representation generated by SBERT proves the influence of sentence context in capturing the semantic relationship between two texts. Compared to traditional methods such as TF-IDF, the proposed model can utilise non-linear features through the integration of FNN, thus enhancing the model's ability to handle texts of varying length and complexity. The SNN architecture allows for identical weights and biases in both sub-networks, thus ensuring consistent comparison between two texts. This overcomes limitations identified in previous studies, such as the inability to capture deep semantic relationships between texts [10, 11].

Tables 2–6 show the model performance evaluation results on three different datasets (MSRP, STS, and QQP). The evaluation was performed with five variants of the SBERT model: 'all-MiniLM-L6-v2', "all-MiniLM-L12-v2", "paraphrase-MiniLM-L6-v2", "paraphrase-MiniLM-L12-v2", and "multi-qa-MiniLM-L6-cos-v1", the number of epochs between 100 and 1000 with multiples of 100, and the learning rate consisting of 0.5, 0.1, 0.05, 0.01, 0.005, and 0.001. The results show that the SBERT model variant, number of epochs, and learning rate greatly affect the model performance. On the STS dataset, the 'all-MiniLM-L6-v2' model achieved the highest accuracy of 87.82% with 400 epochs and a learning rate of 0.5, indicating that it is highly effective in capturing the semantic relationship between two texts. On the QQP dataset, the 'multi-qa-MiniLM-L6-cos-v1' model achieved the highest accuracy of 76.72% with 300 epochs and a learning rate of 0.5. Although this model is more optimised for question and answer tasks, it shows the flexibility of the model in other applications, such as paraphrase detection. While on the MSRP dataset, the 'paraphrase-MiniLM-L12-v2' model achieved the highest accuracy of 73.79% with 200 epochs and a learning rate of 0.5, which is in line with the main purpose of this model in detecting paraphrases.

As shown in Table 7, the best results are achieved with the optimal combination of the SBERT model, number of epochs, and learning rate. The 'all-MiniLM-L6-v2' model gives the best results on the STS dataset with 87.82% accuracy, while the 'multi-qa-MiniLM-L6-cos-v1' model gives the best results on the QQP dataset with 76.72% accuracy. While on the MSRP dataset, the 'paraphrase-MiniLM-L12-v2' model recorded the highest accuracy of 73.79%. All these best results show that the right choice of the SBERT model is crucial in matching the characteristics of the dataset and the task at hand. In addition, the range of number of epochs between 300 to 700 is the optimal point for some of the models used, as in this range the models have learnt important features without overfitting. However, some models also showed good stability at low epoch counts, between 100 and 200, or high epoch counts, between 800 and 1000, depending on the complexity of the dataset and embedding characteristics used.

The proposed SNN-FNN hybrid architecture has several advantages. The utilization of SBERT as the initial representation provides a semantically richer numerical vector than traditional methods. This is evident from the test results, which show an improvement in the accuracy of semantic similarity measures. The combination of SNN with FNN allows the integration of the strengths of both architectures, where SNN to learn the shared representation and FNN captures the non-linear relationship between features. Unlike other approaches, such as the Siamese MaLSTM, which uses ELMO embedding [10], this model integrates SBERT to produce a more effi-

cient and accurate representation of the text, thereby reducing the computational burden without sacrificing performance.

In comparison to [16], which used CNN-LSTM to detect duplicate questions on the Quora platform, this hybrid model shows greater flexibility in handling texts of varying lengths. Although CNN-LSTM performed well on short texts, our hybrid model remains relevant as it can capture more complex semantic relationships through the combination of SNN and FNN.

The problem of the lack of approaches capable of capturing deep semantic relationships between texts as well as the limitations of traditional approaches in handling texts of varying length, such as in [1, 10, 16, 17], is successfully addressed through the combination of SNN and FNN, where SNN allows direct comparison between two texts with identical weight assignment, while FNN adds the ability to capture non-linear relationships between features, which is important in capturing complex semantic nuances, as well as the use of SBERT in the combination of SNN and FNN which allows for deeper context modelling than traditional embedding methods. In addition, the parameters used in this model, such as the number of epochs and learning rate, also impact the performance of the model.

While this hybrid model shows good results, some limitations need to be noted. The performance of the model is highly dependent on the quality of the text representation generated by SBERT, where texts with complex or ambiguous context, the quality of this representation may not be optimal, so integration with a larger transformer model can improve the quality of the semantic representation of the text. This model does not integrate an attention mechanism, which may limit the ability of the model to capture important features in long texts, so the implementation of an attention mechanism can improve the sensitivity of the model to a wider context. In addition, the dataset used is limited to a particular domain, so the generalization of the model into other domains, such as scientific, legal, or medical texts, still needs to be further evaluated to ensure the validity of the application in real-world scenarios.

Besides having limitations, this study also has some shortcomings. The model is still sensitive to parameter selection, such as learning rate and number of epochs, which requires further optimization. Integrating with metaheuristic techniques for parameter automation could be a step forward in reducing the dependency on manual parameter selection. In addition, this hybrid architecture has a higher implementation complexity than the basic approach, which can be challenging in practical applications, especially in systems with computational resource constraints.

To overcome the aforementioned limitations and shortcomings, several development steps can be taken. The addition of attention mechanisms into the hybrid architecture to improve the model's ability to capture important features in longer and more complex texts. The use of modern, larger model transformers to improve the semantic representation of text, especially in complex contexts. Application of metaheuristic techniques to automate model parameter selection to reduce model sensitivity to manual parameter selection and improve model training efficiency. In addition, it is possible to evaluate testing on datasets from new domains to validate the generalisability of the model or test the model in real-world applications to ensure relevance and efficiency in practical scenarios.

This study successfully developed a hybrid architecture of SNN with FNN that can measure the semantic similarity of texts compared to traditional approaches. The combination of SNN with FNN, supported by text representation with SBERT, can capture semantic relationships from long texts. However, there are still some areas that need further development, including the integration of attention mechanisms, automatic parameter optimization, and evaluation in new domains. With these development steps, this hybrid model has the potential to become a more robust and flexible solution in various NLP applications.

7. Conclusions

1. Integration of textual representation into a hybrid architecture that combines SNN with FNN to measure semantic text similarity was successfully achieved by utilising SBERT in the hybrid model. The model is developed to capture deep semantic relationships between text pairs by utilising the strengths of both architectures, where SNN is used to compare two texts and FNN to capture non-linear relationships between features. The use of SBERT as the initial representation can consider the sentence context in depth, resulting in semantically rich numerical vectors. In addition, parameter variations such as the number of epochs and learning rate are optimised to ensure stable and efficient model performance. This hybrid architecture shows flexibility in handling texts of varying lengths and overcomes the limitations of traditional approaches.

2. A performance evaluation of the SNN with FNN hybrid architecture for measuring semantic text similarity was conducted on several datasets, showing that it achieved the highest accuracy of 87.82% using the 'all-MiniLM-L6-v2' model on the STS dataset, 76.72% with the 'multi-qa-MiniLM-L6-cos-v1' model on the QQP dataset, and 73.79% using the 'paraphrase-MiniLM-L12-v2' model on the MSRP dataset. This performance shows that the hybrid model has good generalization ability across multiple domains. Parameter optimization also shows that the optimal number of epochs is generally in the range between 300 and 700, while the optimal learning rate is in the range between 0.01 and 0.1. These results prove that this hybrid architecture is not only effective in improving accuracy but also relevant for practical applications in various real-world scenarios.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

The study was performed without financial support.

Data availability

Manuscript has associated data in a data repository.

Use of artificial intelligence

The authors have used artificial intelligence technologies within acceptable limits to provide their own verified data, which is described in the research methodology section.

References

1. Jinarat, S., Pruengkarn, R. (2024). Enhancing Short Text Semantic Similarity Measurement Using Pretrained Word Embeddings and Big Data. 2024 5th International Conference on Big Data Analytics and Practices (IBDAP), 63–66. https://doi.org/10.1109/ibdap62940.2024.10689695
2. Abdalgader, K., Matroud, A. A., Hossin, K. (2024). Experimental study on short-text clustering using transformer-based semantic similarity measure. PeerJ Computer Science, 10, e2078. https://doi.org/10.7717/peerj-cs.2078
3. Alnajem, N. A., Binkhonain, M., Shamim Hossain, M. (2024). Siamese Neural Networks Method for Semantic Requirements Similarity Detection. IEEE Access, 12, 140932–140947. https://doi.org/10.1109/access.2024.3469636
4. Fan, A., Wang, S., Wang, Y. (2024). Legal Document Similarity Matching Based on Ensemble Learning. IEEE Access, 12, 33910–33922. https://doi.org/10.1109/access.2024.3371262
5. Bao, W., Dong, J., Xu, Y., Yang, Y., Qi, X. (2024). Exploring Attentive Siamese LSTM for Low-Resource Text Plagiarism Detection. Data Intelligence, 6 (2), 488–503. https://doi.org/10.1162/dint_a_00242
6. Goltaji, M., Abbaspour, J., Jowkar, A., Fakhrahmad, S. M. (2023). Comparison of text-based and linked-based metrics in terms of estimating the similarity of articles. Journal of Librarianship and Information Science, 56 (3), 760–772. https://doi.org/10.1177/09610006231165759
7. Korade, N. B., Salunke, M. B., Bhosle, A. A., Kumbharkar, P. B., Asalkar, G. G., Khedkar, R. G. (2024). Strengthening Sentence Similarity Identification Through OpenAI Embeddings and Deep Learning. International Journal of Advanced Computer Science and Applications, 15 (4). https://doi.org/10.14569/ijacsa.2024.0150485
8. Reimers, N., Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). https://doi.org/10.18653/v1/d19-1410
9. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv. https://doi.org/10.48550/arXiv.1810.04805
10. Altamimi, A., Umer, M., Hanif, D., Alsubai, S., Kim, T.-H., Ashraf, I. (2024). Employing Siamese MaLSTM Model and ELMO Word Embedding for Quora Duplicate Questions Detection. IEEE Access, 12, 29072–29082. https://doi.org/10.1109/access.2024.3367978
11. Li, R., Cheng, L., Wang, D., Tan, J. (2023). Siamese BERT Architecture Model with attention mechanism for Textual Semantic Similarity. Multimedia Tools and Applications, 82 (30), 46673–46694. https://doi.org/10.1007/s11042-023-15509-4
12. Chicco, D. (2020). Siamese Neural Networks: An Overview. Artificial Neural Networks, 73–94. https://doi.org/10.1007/978-1-0716-0826-5_3
13. Wong, K., Dornberger, R., Hanne, T. (2022). An analysis of weight initialization methods in connection with different activation functions for feedforward neural networks. Evolutionary Intelligence, 17 (3), 2081–2089. https://doi.org/10.1007/s12065-022-00795-y
14. Harumy, T. H. F., Zarlis, M., Lydia, M. S., Efendi, S. (2023). A novel approach to the development of neural network architecture based on metaheuristic protis approach. Eastern-European Journal of Enterprise Technologies, 4 (4 (124)), 46–59. https://doi.org/10.15587/1729-4061.2023.281986
15. Han, S., Shi, L., Tsui, F. (Rich). (2025). Enhancing semantical text understanding with fine-tuned large language models: A case study on Quora Question Pair duplicate identification. PLOS ONE, 20 (1), e0317042. https://doi.org/10.1371/journal.pone.0317042
16. Faseeh, M., Khan, M. A., Iqbal, N., Qayyum, F., Mehmood, A., Kim, J. (2024). Enhancing User Experience on Q&A Platforms: Measuring Text Similarity Based on Hybrid CNN-LSTM Model for Efficient Duplicate Question Detection. IEEE Access, 12, 34512–34526. https://doi.org/10.1109/access.2024.3358422
17. Abdalla, H. I., Amer, A. A. (2021). Boolean logic algebra driven similarity measure for text based applications. PeerJ Computer Science, 7, e641. https://doi.org/10.7717/peerj-cs.641
18. Xue, Y., Tong, Y., Neri, F. (2022). An ensemble of differential evolution and Adam for training feed-forward neural networks. Information Sciences, 608, 453–471. https://doi.org/10.1016/j.ins.2022.06.036
19. Hemeida, A. M., Hassan, S. A., Mohamed, A.-A. A., Alkhalaf, S., Mahmoud, M. M., Senjyu, T., El-Din, A. B. (2020). Nature-inspired algorithms for feed-forward neural network classifiers: A survey of one decade of research. Ain Shams Engineering Journal, 11 (3), 659–675. https://doi.org/10.1016/j.asej.2020.01.007
20. Harumy, T. H. F., Zarlis, M., Effendi, S., Lidya, M. S. (2021). Prediction Using A Neural Network Algorithm Approach (A Review). 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM), 325–330. https://doi.org/10.1109/icsecs52883.2021.00066
21. Saini, J. R., Vaidya, S. (2024). A Novel Page Similarity Classification Algorithm for Healthcare Web URL Classification. Proceedings of Third International Conference on Computing and Communication Networks, 291–301. https://doi.org/10.1007/978-981-97-2671-4_22
22. Diamzon, J., Venturi, D. (2025). Uncertainty Propagation in Feed-Forward Neural Network Models. https://doi.org/10.2139/ssrn.5201857
23. Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press. Available at: https://www.deeplearningbook.org/
24. Shen, Y., Lin, Z. (2024). PatentGrapher: A PLM-GNNs Hybrid Model for Comprehensive Patent Plagiarism Detection Across Full Claim Texts. IEEE Access, 12, 182717–182725. https://doi.org/10.1109/access.2024.3508762