International Journal of Basic and Applied Sciences, 14 (3) (2025) 150-155

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/vnjbkz65
Research paper



Afaan Oromoo Textual Entailment Classification Using Deep Learning Approach

Diro Tolosa [1] *, Arulmurugan Ramu [2], Ramata Mosissa [3], Teshome Debushe [3], Desalegn Tasew [4], Diriba Gichile [5]

[1] Department of Educational Technology and Information Management, Mattu University
[2] Department of Computational Sciences and Software Engineering, Heriot-Watt International Faculty, K. Zhubanov Aktobe Regional University
[3] Department of Information Technology, Mattu University
[4] Department of Electrical and Computer Engineering, Mattu University
[5] Department of Computer Science, Mattu University
**Corresponding author E-mail: tolosadiro100@gmail.com*

Received: June 7, 2025, Accepted: July 15, 2025, Published: July 20, 2025

Abstract

Natural language processing (NLP) is the field that enables computers to understand and use human language. Textual entailment—a key NLP task— determines if a hypothesis can logically follow from a given premise. As we reviewed, the model designed and developed for other languages is not used for Afaan Oromoo textual entailment classification, as its semantics and syntax are different when compared with other languages. To address the gap, we proposed an Afaan Oromoo textual entailment classification model. We used Support Vector Machine (SVM) as a baseline to compare with three deep learning architectures: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BiLSTM) by comparing their performance to identify the most effective approach with fasttext and word2vec word embedding. We collected a dataset of 13,060 sentence pairs in Afaan Oromoo. The accuracy of SVM was 55.82% and the accuracy of CNN, LSTM, and BiLSTM was 72.8%, 75.57% and 80.47% respectively, with fasttext word embedding. Considering the limited resources available for Afaan Oromoo NLP, the result is encouraging. As a starting point, this study offers a basis for additional investigation and advancement in this field and contributes to the development of Afaan Oromoo's Natural Language Processing capabilities.

Keywords: *Afaan Oromoo; Deep Learning; Natural Language Processing; Textual Entailment Classification.*

1. Introduction

Humans can communicate with each other using some form of language, either by text or speech. They can exchange the message in their natural language without any trouble. But, to make interaction with computers, the language that human beings use should be understandable by computers. This aims to enable computers to read and comprehend the natural language used by humans [1]. The ability of computer software to understand spoken and written human language is known as Natural Language Processing. The main goal is to teach computers to understand, comprehend, analyze, and manipulate human languages. Nowadays, many of the programs we use daily are based on machines' capacity to understand human language [2]. Natural language allows for several meanings to be stated by a single text as well as multiple texts that express the same meaning, indicating a many-to-many mapping relationship between the meanings and phrases of language. Theoretically, a comprehensive semantic interpretation into a logic-based representation of a text's meanings would be necessary for accurate interpretation [3]. The evaluation of Natural Language Processing systems and the question of whether a model performs better at generating text or understanding language is becoming more crucial as NLP technologies become more extensively used [4]. The way computers understand the structure and meaning of human language is called Natural language understanding. Here, understanding the meaning of a given piece of text is an open problem in NLP [5].

Text Entailment is a branch of Natural Language Processing that focuses on understanding the meaning of sentences. It involves determining the relationship between pairs of texts. We can classify the textual entailment into: Entailment, Contradiction, and Neutral[5]. The label between two sentences is Entailment when sentence one supports sentence two. The label between two sentences is a Contradiction when sentence one contradicts sentence two. The label between two sentences is Neutral when there is no correlation between the two pairs of sentences. The aim of textual entailment recognition (RTE) is to ascertain if the hypothesis can be reasonably deduced from the premise given a pair of sentences. In 2005, the significance of Textual Entailment Recognition became a generic task that addresses the main requirements for semantic inference in a wide range of applications using natural language processing. Many NLP applications [6], such as Question Answering, Information Extraction, text summarization, and Machine translation evaluation, need automation of

Copyright © Diro Tolosa et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

textual entailment in order to recognize that a particular target meaning can be inferred from different text variants[7]. So, textual entailment recognition (RTE) has been a very popular research area in the last years and currently also. Recognizing Textual Entailment (RTE) has gained popularity in the NLP community since its inception in 2005 because it appears to serve as a common framework for analyzing, contrasting, and assessing various approaches used in NLP applications to address semantic inference, a problem that many NLP applications share.

Different approaches are developed for Textual Entailment Recognition. Most of the Recognizing Textual Entailment systems done are based on Machine Learning, lexical, or semantic approaches. These approaches use features such as lexical, syntactic, and semantic features. The lexical approach works directly on the input surface strings, but it performs poorly on the task of recognizing textual entailment. Because textual entailment is a directional relation, where the text contains more information than the hypothesis. Another known approach to recognizing textual entailment is based on which syntactic information is usually represented as a directed graph [3].

The rapid development of deep learning with natural language processing brings hope for possibly alleviating the problem of avoiding manual feature engineering and learning semantic representations for natural language [8]. Deep learning models have achieved impressive performance on various Natural language inference tasks. They leverage neural networks to learn complex patterns and representations from a large amount of text data [8]. In this study, we focused on comparing traditional machine learning with the deep learning architectures developed for Afaan Oromoo language textual entailment classification.

The work's contributions are:

- We prepare the Afaan Oromoo textual entailment dataset
- We develop an Afaan Oromoo textual entailment classification model
- We use fasttext and word2vec for comparison of word embedding.
- We use SVM with TF-IDF as a baseline for comparing with BiLSTM, LSTM, and CNN.
- We use dropout with BiLSTM, LSTM, and CNN models for sentence embedding
- We concatenate the embedded sentences

2. Related works

With a Qubee writing system, Afaan Oromoo is a Cushitic language that is mostly spoken in Ethiopia and its surrounding nations. The language is renowned for having a complex morphology that incorporates several inflectional processes. Simple clauses, or SOV word-order typologies, have the subject (s) at the beginning, the object (O) or complement in the middle, and the verb (V) at the end[9]. As the Afaan Oromoo language has its own syntax and semantic rules, the model developed for other languages' textual entailment cannot work for it.

Helina Mesfin[10], developed “Amharic textual entailment classification by using a deep learning approach”. She used a reverse side sentence matching model with five main layers. Word embedding, sentence embedding, sentence matching, aggregation, and prediction layers are the layers used. In the sentence embedding layer, she used Bi-LSTM to remember long sentence sequences of the dataset. A pair of 8700 sentences is used for Amharic textual entailment, and the model produced a scoring of 77% accuracy for the training dataset and 72% accuracy for the test dataset.

Han et al. [11]In their paper entitled “Recognizing Textual Entailment using Deep Learning Techniques,” they used a character-level convolutional network to replace the standard word-level embedding layer, and they used the intra-attention layer to capture the intra-sentence semantics. They demonstrated the CIAN model, a sequence encoder with extensive semantic and orthographic properties that can encode lengthy sentences at the character level. Furthermore, the attention mechanism makes the model highly interpretable, making it possible for users to comprehend how the model operates. In the matched test set, the model's accuracy was 90%, and in the mismatched test set, it was 60%.

Lyu et al. [12]They proposed a novel two-step procedure to recognize textual entailment. In order to learn the joint representation of the text-hypothesis pairings, they first constructed a joint layer of Restricted Boltzmann Machines (RBM). Next, to identify textual entailment for each pair, the reconstruction error is computed by comparing the original representation with the rebuilt representation obtained from the joint layer. A sizable news corpus is automatically used to create the joint RBM training data. The findings of the experiment demonstrate how the concept affected textual entailment performance, with an accuracy of 67.5% for training pairs and 66.5% for test pairs.

According to Haggag et al. [13]They proposed a “hybrid approach” which is based on lexical, syntactic, and semantic analysis and a Deep Learning classifier entailment module. The labeled texts and hypothesis pairs provided by the Text Analysis Conference (TAC) are used. By training the model with 3567 samples and testing the model with 1000 samples, it showed 89.8% accuracy.

Another method applied to the textual entailment recognition is a lexical and syntactic features hybrid method. Partha et al. [14]In their paper entitled “A Hybrid Textual Entailment System using Lexical and Syntactic Features,” they have described a two-way textual entailment recognition system that uses lexical and syntactic features. They used a hybrid textual entailment system that is Support Vector Machine (SVM) based. They developed the lexical and syntactic hybrid system utilizing 2400 text-hypothesis pairs and a collection of test gold, development, and annotated sets. The test set evaluation scores revealed that for YES decisions, there was 55.30% precision and 58.40% memory, and for NO decisions, there was 55.93% precision and 52.80% recall. The semantic rules have not been taken into account.

Dwijen et al. [15]In their paper, they have investigated the idea of “Partial Textual Entailment for Indian social media text (SMT)”. They created an annotated corpus for Bengali tweets using Partial Textual Entailment and suggested a Sequential Minimal Optimization (SMO) based method. They employed the recognition technique known as Partial Textual Entailment, or PTE. Their approach was evaluated using experiment findings, and the accuracy of the F measure was 98.3%.

Quang Nhat et al. [16]They presented a “machine learning based textual entailment recognition system for Japanese language”. To learn the entailment classifier, they are tagged to a binary class, and various entailment features taken from original Japanese pairs and their English translation are combined. Among the participant groups, they achieved a 58% accuracy rate in the Japanese Binary class subtask on the test set. Their research suggested that the RTE system's performance could be enhanced via machine translation.

Mariam et al. [17], in their paper entitled “Textual Entailment for Arabic Language based on Lexical and Semantic Matching”, used a lexical analysis technique of Textual Entailment to study the suitability of the technique for the Arabic language. Furthermore, a semantic matching technique was incorporated to improve the suggested entailment system's accuracy. Word overlap calculations, bigram extraction, and matching provide the foundation of the lexical analysis. To improve word matching accuracy, semantic matching has been

combined with word overlap. Using a binary classification, the system was able to classify the relationship between pairs of sentences with a recall of 61% and a precision of 68%, 58%, and 58% for Entails and NotEntails, respectively.
Nada et al. [18], In their paper, they addressed the problem of textual entailment in Arabic. They made use of distributional representations in addition to traditional features. When compared to other fundamental surface-level matching features, they have demonstrated that the employment of word representation-based features yields a satisfactory outcome. The size of the set they assessed was constrained. Lastly, there are still issues with the current system, such as the different methods word embedding could be applied. Their method produced 76.2% accuracy.
Mabrouka et al. [19]They developed a "semantic method for recognizing textual entailment in Arabic". The directional semantic entailment relationship between text/hypothesis pairings is detected by the model. To be more precise, they concentrated on work at the sentence level, using word sense disambiguation and semantic similarity measures to identify entailment relationships in the context of the Arabic question/answer system. They achieved a 70% accuracy rate.
From the review of the available studies, most of the research has been conducted on highly resourced languages like English, Arabic, and the local language Amharic. So, we proposed Afaan Oromoo textual entailment classification to fill the gap.

3. Methodology

In this work, we used Design Science Research. The work starts with data collection. In the deep learning process, a huge amount of data is required to perform the required activity. Because deep learning performs best with larger corpora, the quantity of datasets collected influences the model's performance.

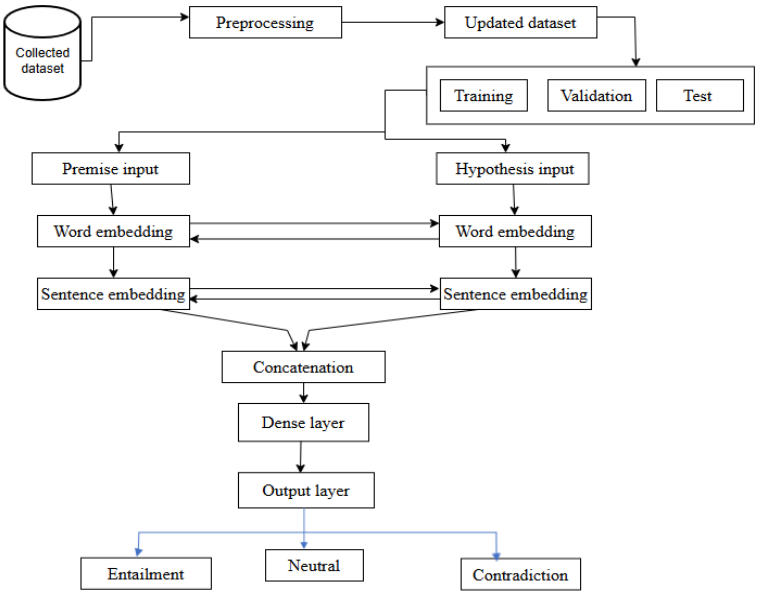


Fig. 1: Afaan Oromoo Textual Entailment Classification Architecture.

For this work, the dataset that includes pairs of sentence one and corresponding sentence two labeled with their entailment relationship has been gathered from different Afaan Oromoo texts (13000 pair of sentences), like: Newspaper, Websites and others and some of the datasets (60 pair of sentences) has been translated from SNLI dataset (English Language dataset). The translated dataset hasn't shown biases, because the sentences about proverbs, pottery, and multi-sentence inference are not included. With the help of Afaan Oromoo experts, the collected data (13060 pairs of sentences) is annotated with its appropriate labels. After annotation data succeeds, 80% of the dataset is used for training, 10% is used for testing, and 10% for validating the model.
As shown in Figure 1, the general architecture of Afaan Oromoo textual entailment using a deep learning approach consists of: Word embedding layer, Sentence embedding layer, Concatenation layer, Dense and dropout layer, and Output layer. The annotated dataset is to be preprocessed with the preprocessing algorithms. Then, word embedding and sentence embedding techniques are used to convert the updated dataset into numerical vectors. The dataset becomes available for building the model. Two embedding techniques were used: Word2Vec and FastText with similar hyperparameters. Three neural architectures (CNN, LSTM, BiLSTM) were used. At last, the final hidden states are concatenated to combine the information from both sentences. The combined representation is then used to determine the relationship between the two sentences. To avoid overfitting, the concatenated vector is passed through two layers: a dense layer and a dropout layer. Finally, probabilities for each class—Entailment, Neutral, and Contradiction—are generated by an output layer using a softmax activation function. All architectures (CNN, LSTM, and BiLSTM) included an embedding layer, sentence encoder, concatenation layer, dense layers, and a softmax output layer. The following hyperparameters are set for the three deep learning architectures.

- Words are transformed into 300-dimensional vectors.
- Models are trained in batches of 32 or 64 samples at a time, and they are trained using the Adam optimizer.
- A Softmax function is used to process the output and predict one of three probable classes.
- The system calculates error using categorical cross-entropy and dropout to prevent overfitting.
- Training takes place across 10 or 20 rounds, or epochs.

Table 1: Summary of Hyperparameters Used

Models	Hyper parameter	Value
CNN	Activation function for output	Softmax function
LSTM	Optimizer	Adam
BiLSTM	Batch size	32, 64

Embedding dimension	300
Number of units in LSTM hidden units	128
Pool-size in MaxPooling	2
Dropout rate	0.5
Number of units in Dense layer	64
Number of classes in output layer	3
Activation function in dense layer	ReLu
Loss function	Categorical cross-entropy
Learning rate	0.001
Number of epochs	10, 20

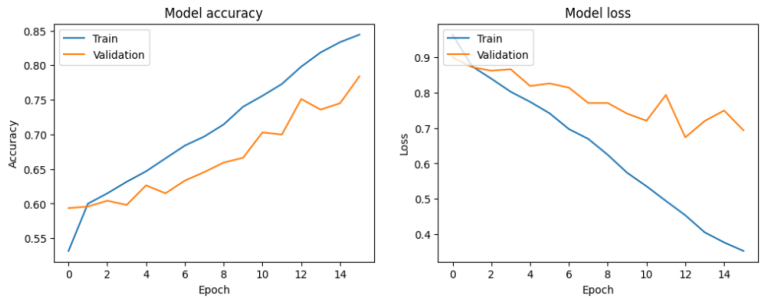


Fig. 2: Bilstm with Word2vec with Epoch 20 and 32 Batch Size.

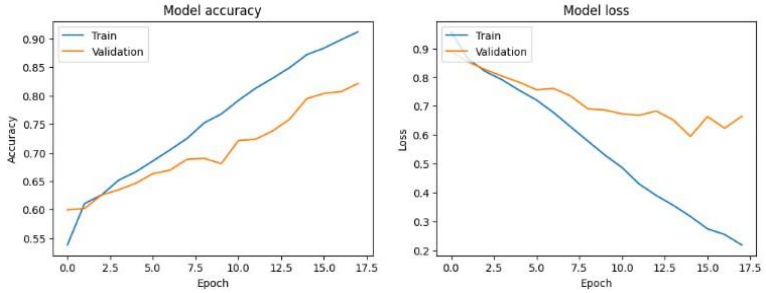


Fig. 3: Bilstm with Fasttext with 20 Epoch and Batch Size 32.

4. Results and discussion

For our work, we used a Support Vector Machine (SVM) as a baseline to highlight the advantages of deep learning architectures (CNN, LSTM, and BiLSTM). For deep learning architectures, different scenarios have been considered during the experiment by using both fasttext and word2vec for word embedding.

a) Batch size 64 with 10 epochs
b) Batch size 32 with 10 epochs
c) Batch size 64 with 20 epochs
d) Batch size 32 with 20 epochs

When we get to the discussion section, the following is a summary of all the models together with the corresponding epochs, batch size used, learning rate, training accuracy, validation accuracy, and test accuracy.

Table 2: Summary of SVM by Using TF-IDF

Model	Training dataset Accuracy	Validation dataset Accuracy	Test dataset Accuracy
SVM	71.90%	55.67%	55.82%

Table 3: Summary of Deep Learning Models' Accuracy with Fasttext

Model	Epoch	Batch size	Learning rate	Training dataset accuracy	Validation dataset accuracy	Test dataset accuracy
CNN	10	32	0.001	65.01%	63.23%	66.04%
	10	64	0.001	66.06%	65.01%	65.54%
	20	32	0.001	72.83%	68.98%	72.8%
	20	64	0.001	73.3%	69.06%	71.9%
LSTM	10	32	0.001	75.44%	69.75%	70.29%
	10	64	0.001	71.75%	63.9%	69.52%
	20	32	0.001	87.25%	77.79%	75.57%
	20	64	0.001	85.59%	75.57%	72.43%
BiLSTM	10	32	0.001	75.20%	69.06%	70.21%
	10	64	0.001	73.03%	65.62%	68.98%
	20	32	0.001	91.23%	82.15%	80.47%
	20	64	0.001	89.74%	80.70%	78.79%

Table 4: Summary of Models with Word2vec

Model	Epoch	Batch size	Learning rate	Training dataset accuracy	Validation dataset accuracy	Test dataset accuracy
CNN	10	32	0.001	65.82%	63.39%	64.77%
	10	64	0.001	66.52%	63.85%	64.85%
	20	32	0.001	73.87%	68.45%	70.21%
	20	64	0.001	74.81%	70.59%	73.58%
LSTM	10	32	0.001	72.09%	65.54%	66.92

BiLSTM	10	64	0.001	68.98%	62.94%	64.93%
	20	32	0.001	85.52%	74.40%	74.12%
	20	64	0.001	82.22%	71.66%	70.64%
	10	32	0.001	73.74%	66.15%	66.30%
	10	64	0.001	70.32%	63.24%	64.31%
	20	32	0.001	84.42%	78.42%	74.57%
	20	64	0.001	83.39%	73.96%	70.29%

Table 5: Comparison of SVM and BiLSTM

Model	Training dataset Accuracy	Validation dataset Accuracy	Test dataset Accuracy
SVM	71.90%	55.67%	55.82%
BiLSTM	91.23%	82.15%	80.47%

Table 2 shows the accuracy of the training dataset, validation dataset, and test dataset of SVM with TF-IDF vectorization to represent the combined premise and hypothesis pairs. Table 3 shows the accuracy of the training dataset, validation dataset, and test dataset of CNN, LSTM, and BiLSTM with the experiment hyperparameters and fasttext word embedding. Table 4 shows the accuracy of the training dataset, validation dataset, and test dataset of CNN, LSTM, and BiLSTM with the experiment hyperparameters and word2vec word embedding. Table 5 shows the comparison of SVM and BiLSTM with fasttext word embedding. In order to evaluate the classification performance on Afaan Oromo text data, the Support Vector Machine (SVM) classifier was used as a conventional machine learning baseline. The SVM model showed dependable performance in classifying text into the designated categories using TF-IDF vectorization to represent the combined premise and hypothesis pairs. It was a good option for preliminary testing due to its efficiency in managing high-dimensional sparse feature spaces, like those generated by TF-IDF. Particularly on smaller datasets where deep learning models like BiLSTM are prone to overfitting or demand a lot of processing power, the model produced competitive results. SVM offered a straightforward, quick, and understandable method, even though it was unable to capture word order or contextual dependencies between words. Its overall accuracy, however, was inferior to that of the BiLSTM model, particularly when handling intricate sentence structures where sequence information was crucial. Consequently, it was clear that deep learning architectures, which learn from contextual embeddings and word sequences, provided better performance for more linguistically complex tasks, even though SVM was a solid baseline for text classification tasks in Afaan Oromo. In Table 5, we found that SVM scored less than the three deep learning architectures. BiLSTM model with fasttext word embedding performs better than other SVM, CNN, and LSTM With 91.23% training accuracy, 82.15% validation accuracy, and 80.47% test dataset accuracy, the BiLSTM model outperformed the other models.

The fact that a single text can have multiple interpretations and those distinct texts can have the same meaning is one of the difficulties with natural language inference. One of the features of natural language is the variety of ways it might convey an idea. We have therefore performed error analysis based on Afaan Oromoo's various sentence forms, keeping these notions in mind. Here, the following Table 6 describes more.

Table 6: Error Analysis between Models

Types of Sentences	Model	Predicted label	Actual label
Paraphrasing sentences	CNN	Entailment	
Premise: Ijoolleen kolfaa jiru.	LSTM	Entailment	Entailment
Hypothesis: daa'imman ganmadaa jiru	BiLSTM	Entailment	
Long sentences	CNN	Entailment	
Premise: Murteessaan tokko dirree tokko irratti taphattoota tokko tokko waliin haasa'aa jira.	LSTM	Entailment	Entailment
Hypothesis: abbaan murtii rukuttaa adabbii kennan irratti taphattootaa waliin mari'achaa jiru	BiLSTM	Entailment	
Conditional sentences	CNN	Neutral	
Premise: Garee saawwanii laga tokko keessa dhaabbatan.	LSTM	Contradiction	Contradiction
Hypothesis: saawwan dirree keessa fiigaa jiru.	BiLSTM	Contradiction	
Active/Passive sentences	CNN	Neutral	
Premise: namoonni mana ijaaruudhaaf simintoo fayyadamu.	LSTM	Neutral	Neutral
Hypothesis: simmintoon mana ijaaruuf ni fayyada	BiLSTM	Neutral	
Quantifier words	CNN	Contradiction	
Premise: Saroonni lama ala jiru.	LSTM	Contradiction	Contradiction
Hypothesis: Saroonni afur mana keessa jiru.	BiLSTM	Contradiction	

5. Conclusion and recommendation

We performed a comparison analysis with Support Vector Machine (SVM) and three deep learning architectures: CNN for Afaan Oromoo textual entailment classification, LSTM for Afaan Oromoo textual entailment classification, and BiLSTM for Afaan Oromoo textual entailment classification. We have collected the dataset manually from different Afaan Oromoo texts, and, we translated some of the dataset from the Standard Natural Language Inference dataset available for English. SVM is the baseline we used to highlight the advantage of deep learning architectures. CNN, LSTM, and BiLSTM are the three deep learning models that we used to train the dataset. For each all we set the hyperparameters, we used varying epochs and batch sizes during training, and we noted variations in test, validation, and training accuracy. We carried out error analysis on various sentence types, including conditional, long, active/passive, and paraphrase sentences, in order to characterize how the model handles Afaan Oromoo sentence structures. Our work has room for future work. Other researchers can use hybrid models (e.g., combining lexical features with deep learning), can work on noisy texts, can work on sentences containing poetry, proverbs, different types of sentences, and multi-sentence inference, can explore transformer-based models like BERT fine-tuned for Afaan Oromoo, can expand the dataset with more domains and informal text sources, can incorporate attention mechanisms to improve interpretability and classification accuracy, and develop tools and benchmarks for other under-resourced African languages.

Acknowledgement

The authors would like to thank the Department of Information Technology, College of Engineering and Technology of Mattu University for their support. The authors declare no conflict of interest

References

[1] R. Navigli and V. R. Elena, "Natural Language Understanding : Instructions for (Present and Future) Use," pp. 5697–5702, 2003. https://doi.org/10.24963/ijcai.2018/812.
[2] K. M. Verspoor and K. B. Cohen, "Synonyms," no. June, 2018,
[3] M. H. Haggag and A. Mohammed, "Different Models and Approaches of Textual Entailment Recognition," vol. 142, no. 1, pp. 32–39, 2016. https://doi.org/10.5120/ijca2016909667.
[4] A. Poliak, "A Survey on Recognizing Textual Entailment as an NLP Evaluation".
[5] A. Mishra, "Deep Learning Techniques in Textual Entailment".
[6] L. Bentivogli, I. Dagan, H. T. Dang, D. Giampiccolo, B. Magnini, and R. Gan, "The Fifth PASCAL Recognizing Textual Entailment Challenge".
[7] R. Gan, L. P. Group, O. M. Way, H. Language, and F. B. Kessler, "Recognizing textual entailment : Rational , evaluation and approaches," vol. 15, no. 4, 2009, https://doi.org/10.1017/S1351324909990209.
[8] P. Eleftheriadis, I. Perikos, and I. Hatzilygeroudis, "Evaluating Deep Learning Techniques for Natural Language Inference," *Appl. Sci.*, vol. 13, no. 4, 2023, https://doi.org/10.3390/app13042577.
[9] E. T. Garoma, "Demonstratives in Afaan Oromoo," *Cogent Arts Humanit.*, vol. 11, no. 1, p., 2024, https://doi.org/10.1080/23311983.2023.2297494.
[10] H. Mesfin, "College of Natural Sciences Helina Mesfin Advisor : Fekade Getahun (PhD)," 2020.
[11] M. R. Costa-jussà, "Recognizing Textual Entailment using Deep Learning Techniques Supervisor :," 2017.
[12] C. Lyu, Y. Lu, D. Ji, and B. Chen, "Deep Learning for Textual Entailment Recognition," 2015, https://doi.org/10.1109/ICTAI.2015.35.
[13] M. H. Haggag and A. M. Ahmed, "Recognizing Textual Entailment based on Deep Learning Approach," vol. 181, no. 43, pp. 36–41, 2019.
[14] P. Pakray, S. Bandyopadhyay, and A. Gelbukh, "A Hybrid Textual Entailment System using Lexical and Syntactic Features". https://doi.org/10.5120/ijca2019918515.
[15] D. Rudrapal, A. Das, and B. Bhattacharya, "Recognition of Partial Textual Entailment for Indian Social Media Text," vol. 23, no. 1, pp. 143–152, 2019, https://doi.org/10.13053/cys-23-1-2816.
[16] Q. Nhat and M. Pham, "A Machine Learning based Textual Entailment Recognition System of JAIST Team for NTCIR9 RITE".
[17] M. Khader, A. Awajan, and A. Alkouz, "Textual Entailment for Arabic Language based on Lexical and Semantic Matching," no. October, 2016, https://doi.org/10.21700/ijcis.2016.109.
[18] N. Almarwani and M. Diab, "Arabic Textual Entailment with Word Embeddings," pp. 185–190, 2017. https://doi.org/10.18653/v1/W17-1322.
[19] M. B. W. Bakari and M. Neji, "Recognizing Textual Entailment for Arabic using seman- tic similarity and Word Sense Disambiguation," vol. 3, no. 1, pp. 1–10.