Journal of Intelligent Systems and Internet of Things
Vol. 16, No. 02, PP. 60-67, 2025
DOI: https://doi.org/10.54216/JISIoT.160205

A Memory Efficient Adversarial Attention Tree-Structured Deep Learning Model for Classification

Nirmala Veluswamy[1,*], Jayanthi Boopathy[2]

[1]Research Scholar and Assistant Professor, School of Computer Studies, RVS College of Arts and Science, Sulur, Coimbatore 641 402, Tamil Nadu, India

[2]Associate Professor, School of Computer Studies, RVS College of Arts and Science, Sulur, Coimbatore 641 402, Tamil Nadu, India

Email: nirmalavelusamy2018@gmail.com; jayanthi@rvsgroup.com

Abstract

The representational and learning power of tree-based deep-learning (DL) classification models makes them a popular choice for dimensional sentiment analysis (DSA). One variant, Tree-structured Convolutional neural network with long short-term Memory (TCL) stands out among many others for its ability to handle uncertainties and unexpected changes in input data while still producing promising Valence-Arousal (VA) predictions for text or image classes. However, the high memory complexity of this model becomes a challenge when dealing with large image/text datasets. To address this issue, this manuscript introduces a Lightweight Adversarial Attention TCL (LAATCL) model for DSA. The proposed model includes a clustering layer in conjunction with the ATCL to decrease memory complexity and enhance performance through reliable sample selection. This model comprises multi-convolution with a clustering layer that utilizes Group-Sparse Non-negative Matrix Factorization (GSNMF) for clustering highly correlated samples. By learning informative and discriminative latent variables across labels, GSNMF helps identify and select samples closest to the cluster centroid for input to the LSTM network, resulting in reduced memory complexity and improved accuracy. The LATCL model outperformed traditional models in experiments conducted on the SST and CIFAR-10 datasets, with accuracies of 93.57% and 95.25%, respectively, demonstrating its usefulness.

Received: January 17, 2025 Revised: February 12, 2025 Accepted: March 06, 2025

Keywords: Tree-based deep learning; Group-Sparse Non-negative Matrix Factorization; Clustering; Dimensional sentiment analysis

1. Introduction

Despite the recent rise in popularity of neural network models, tree-based approaches like Decision Trees and Random Forests are still widely used for metadata jobs. These approaches provide numerous benefits, such as the capacity to manage various attribute classes, resilience to data volume, and ease of feature extraction [1]. Neural network architectures such as CNN are preferred when dealing with spatial information in input data [2]. These architectures can make task-specific assumptions, reducing the need for domain-specific expertise in certain applications like image categorization [3-4]. However, traditional Fully Connected Networks (FCNs) lack inherent biases towards high-dimensional raw data, posing challenges in developing DNNs using tree-based methods [5]. Numerous systems continue to depend on conventional DT learning loops, and there is currently no broadly recognized neural architecture that seamlessly integrates tree-based approaches [6]. This knowledge gap raises questions about the general applicability of neural architectures in various contexts. CNNs excel at identifying objects based on salient characteristics, making them ideal for large-scale image categorization tasks [7].

Supervised training on a large dataset of labeled images is the primary method used to train CNNs to recognize objects. The network learns distinctive features from labeled images over time, gradually acquiring knowledge and adapting to new information as it becomes available. When a segment of the feature space undergoes modification, CNN quickly updates the entire structure by integrating feature extraction and categorization [8].

However, iterative learning poses a risk of permanent information loss. Retrained CNNs need to leverage historical data to update current knowledge. To tackle the issue of catastrophic forgetting and leverage previously obtained characteristics, an adaptive hierarchical network called Tree-CNN (T-CNN) [9] was developed. The T-CNN network is composed of CNNs that grow gradually and acquire new labels. Its hierarchical architecture, extended with feature-sharing-based branching, can accommodate additional labels as nodes. By focusing on leaves grouped into coarse super-classes by early nodes, the network achieves better classification. While the enlarged network can easily incorporate pre-trained convolution layers, it faces challenges in acquiring task-related texts that possess the required attributes. Afterwards, in an effort to improve the level of detail in dimensional sentiment analysis, the T-CNN-LSTM model, abbreviated as TCL, was presented [10] to forecast VA sentence scores by combining regional CNN and LSTM.

The CNN divides the text into segments and allocates weights according to their significance in forecasting VA, whereas the LSTM compiles data to estimate VA for each segment. It employs a region division approach to identify task-related expressions and clauses, considering global text linkages and local information. Although it uses structured data for VA classification, it does not address feature learning complexity, lesser prediction rates, or class changes.

To overcome the above challenges and enhance VA classification efficiency in DSA, this article proposed LAATCL model for DSA. It includes a clustering layer that enhances performance by reducing memory complexity and improving sample selection. This model combines multi-convolution with a clustering layer using the GSNMF to cluster highly correlated samples. GSNMF helps identify and select samples closest to the cluster centroid, improving accuracy and reducing memory complexity in the LSTM network. Here is the structure of the sections that follow: Section II offers a synopsis of the previously published works. In Section III, LAATCL is detailed. In Section IV, we detail the results of the experiments. The investigation is concluded and future enhancements are proposed in the last section.

2. Literature Survey

A few earlier studies on tree-based DL techniques for various purposes are reviewed in this section. He et al. [13] presented a novel hybrid ensemble model to enhance default prediction performance. They utilized Light Gradient Boosting Machine (LightGBM) for learning new feature interactions, CNN for generating deeper feature interactions, and an Inner Product-based Neural Network (IPNN) as a classifier. The ensemble approach combined deep learning and tree-based classifiers for final predictions. However, it had low prediction accuracy and high memory requirements. A Tree-RNN algorithm [14] was used to categorize network traffic more effectively. It uses a binary tree structure with classifiers for classification and traffic splitting rules. Due to unbalanced datasets, the results were not highly reliable.

Zang et al. [15] developed a tree-based ensemble deep learning model called semi-SILDM for O3 prediction by extracting spatiotemporal features from the MODIS data. However, scaling up the data requires additional memory. Aouedi et al. [16] introduced a novel deep-learning approach that combines multiple decision tree-based models in an ensemble. The ensemble includes two levels: base classifiers using decision tree models in the first level, and a deep learning meta-model in the second level to integrate the base classifiers' outputs. However, it has a high memory complexity and challenges in selecting appropriate classifiers for level 1.

Khozeimeh et al. [17] introduced the RF-CNN model for coronary artery disease detection using cardiac magnetic resonance imaging. They utilized CNN to extract relevant information from low-dimensional image copies and integrated these features into DTs for classification. Yet, its high memory consumption remained a limitation despite its enhanced capabilities. In order to improve cloud computing is power efficiency and provide workload predictions, a method called Hierarchical T-CNN (HT-CNN) [18] was created employing sheep flock optimization. . However, the algorithm struggled to adapt to sudden changes in input data, affecting its memory requirements.

Cai et al. [19] developed a tree-structured model that eliminates variance differences among clusters. It automatically constructs a primary classification tree using a clustering technique to group similar subtypes and applies a pruning rule to refine the tree structure. However, it has high memory requirements. Arifuzzaman et al. [20] developed a novel approach that combines DT and DNN to classify nonlinear data. At first, a DT-Based Neural Network (DTBNN) model was designed and then stretched it to DTBDNN with multiple hidden layers. However, the accuracy of the model was low and it was unable to handle an expansion in features.

3. Proposed Methodology

An extensive explanation of the LAATCL model is given in this section. The pipeline for this study includes SFGAN and LAATCL, as shown in Figure 1. Initially, the SFGAN [11] is employed to enhance the training data through the generation of supplementary adversarial samples. Following this, the enhanced data is utilized to train the LAATCL model. The model undergoes evaluation with test data to forecast VA ratings for textual content or to categorize image classes.

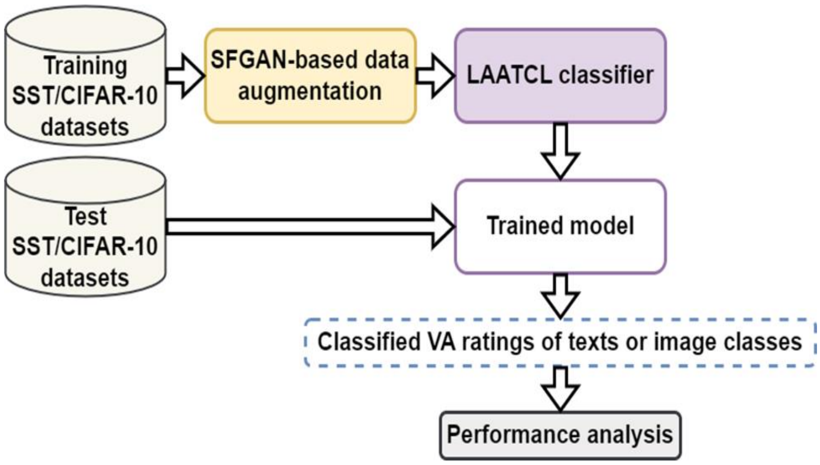

Figure 1. Conceptual Overview of this Study

A. Unsupervised CNN Clustering for sample selection using GSNMF

To reduce the memory requirements, the A Tree-RNN network integrates unsupervised T-CNN with GSNMF clustering as depicted in Figure 2. The network is divided into representation learning (T-CNN) and clustering modules. The representation learning includes unsupervised T-CNN with multiple convolution layers and one clustering layer. It consists of 5 convolutional layers (Conv1 – Conv5) followed by a clustering layer with c clusters. Examine an M-dimensional random vector x characterized by positive or zero components, such as the convolutional features. The vector has N observations denoted as $x_i, i = 1, \ldots, N$, here N represents batch size. The feature matrix is represented as $X = [x_1, \ldots, x_N] \in \mathbb{R}_{\geq 0}^{M \times N}$. NMF aims to find a positive or zero basis matrix $W \in \mathbb{R}_{\geq 0}^{M \times L}$ and a coefficient matrix $H \in \mathbb{R}_{\geq 0}^{L \times N}$ that minimizes the difference between X and the product of W and H, as:

$$X \approx WH \qquad (1)$$

Mostly, the count of latent variables (L) is relatively smaller than the minimum of the dimensions of the input matrix (M, N). Traditional NMF is effective at identifying informative latent variables. However, in large DNNs like CNN and LSTMs, there are often redundant and highly correlated units. To tackle this challenge and identify a set of related CNN features, further group-sparsity constraints are introduced to NMF. The elastic net regularization, which integrates l_1 and l_2-norm penalties, is commonly used for group-sparse regularization. The l_1 penalty promotes sparsity in the model, while the l_2 penalty encourages a smoothing and grouping effect.

By applying a weighted combination of l_1 and squared l_2 penalties to H, CNN can efficiently select features that are correlated with the target data while eliminating those that are uncorrelated. The group-sparsity characteristic facilitates the attainment of the intended group-sparse representations. The objective function resulting from GSNMF is articulated as follows:

$$f(W,H) = \frac{1}{2}\|X - WH\|_F^2 + \frac{\lambda_1}{2}\|H\|_2^2 + \lambda_2\|H\|_1, s.t. W, H \geq \qquad (2)$$

In Eq. (2), λ_1 and λ_2 represent the hyperparameters that control the significance of l_1 and l_2 regularization terms.

(i) Optimization

Incorporating group-sparsity into H, the alternating minimization method and multiplicative updating rule improve Eq. (2). The ideas used in conventional NMF are still applicable to the W update rule. H is optimized by gradient descent, which is subject to a first-order update rule expressed as

$$H \leftarrow H - \eta * \frac{\partial f(H)}{\partial H} \qquad (3)$$

In Eq. (3), $*$ stands for the multiplication of elements one by one, while the matrix η represents the size of the steps. Calculating the derivative of $f(H)$ in Eq. (3) using H as an input yields

$$\frac{\partial f(H)}{\partial H} = -W^T X + W^T W H + \lambda_1 H + \lambda_2 I \qquad (4)$$

In Eq. (4), the symbol I stands for an all-one matrix with the same dimension as H. Eq. (4) is the sub-gradient at 0 since the ll_1-norm is not discriminable at 0. Next, think about the adjustable step size η as:

$$\eta = \frac{H}{W^T W H + \lambda_1 H + \lambda_2 I} \qquad (5)$$

In Eq. (5), the division is element-wise. After that, the below update rule is obtained.

$$\begin{cases} W \leftarrow W * \dfrac{XH^T}{WHH^T} \\ H \leftarrow H * \dfrac{W^T X}{W^T W H + \lambda_1 H + \lambda_2 I} \end{cases} \qquad (6)$$

In Eq. (6), H denotes the novel feature representation. Eq. (6) is a simple change to the typical NMF optimization's multiplicative update rule. Since the update rules are multiplicative, the non-negativity of W and H is preserved if they are initialized as non-negative.

(ii) Unsupervised Fine-Tuning of CNN Using GSNMF

Figure 2 shows that, in contrast to supervised fine-tuning, which makes use of the conventional cross-entropy loss, unsupervised fine-tuning makes use of the data instance reconstruction loss from Eq. (2). Typically, the CNN stays frozen during training and the GSNMF takes over. When the fine-tuning scenario is in play, the GSNMF gets to work improving the CNN model. Consistent with conventional NMF approaches, this research uses extra l_2 normalization layers on X and the basis matrix W before the factorization layer. Normalizing the feature vector norm is a widely adopted approach in unsupervised learning, aimed at avoiding degenerate solutions and preventing the collapse of networks.

The CNN activations X are retrieved from the mini-batch in every iteration after forward propagation. The value of N=128 selected as the mini-batch scale. The update rule in Eq. (6) is used to acquire W and H initially. A conventional Euclidean loss, denoted as $\|X - WH\|_F^2$, is minimized when W and H are determined in Eq. (2). For the CNN representation, the variables are fine-tuned via back-propagation of the Euclidean error.

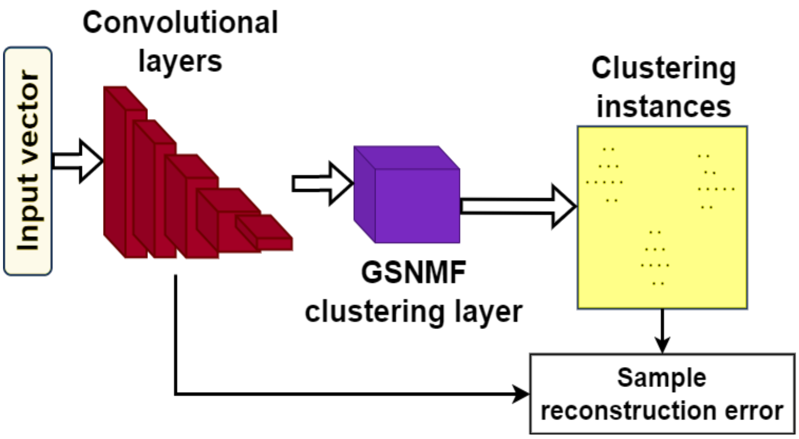

Figure 2. Diagram of Unsupervised Fine-Tuning of the CNN Using GSNMF

B. Dimensional Sentiment Analysis

After obtaining the clustering results, the samples closest to the cluster centroid are identified as significant instances and selected. These instances are then fed into a sequential layer, such as an LSTM network, followed by an attention mechanism and a softmax classifier for final classification. The architecture of LAATCL model is depicted in Figure 3.

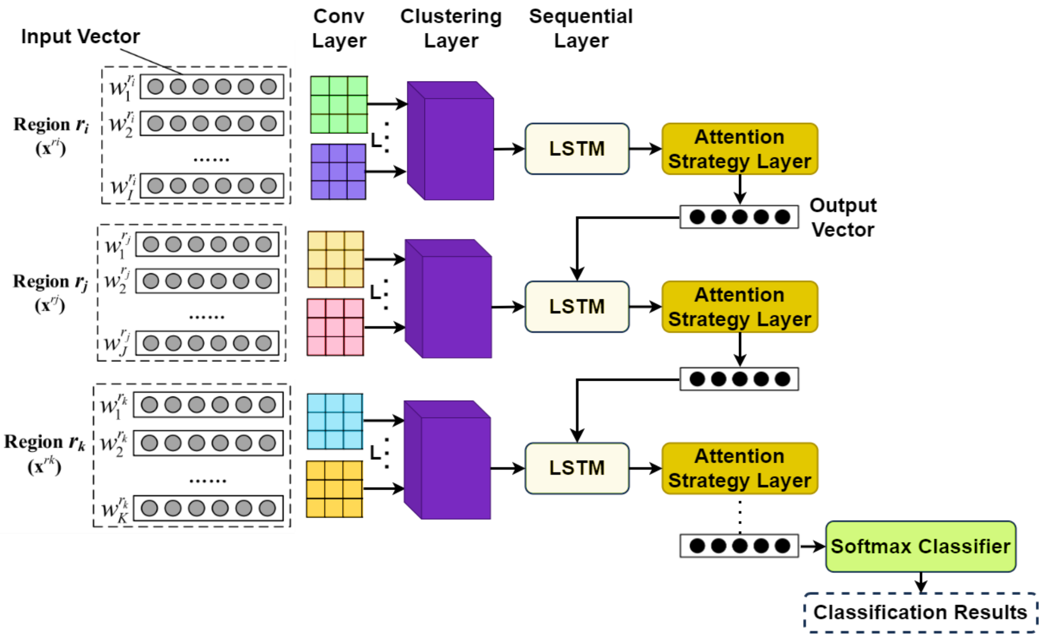

Figure 3. Architecture of LAATCL model

In order to optimize the LAATCL model during training, the mean squared error between the actual label y and the predicted class label $\hat{y}$ is minimized. The model parameters is refined using a backpropagation approach that employs an Adam optimizer. As a result, in order to classify images for dimensional sentiment analysis and VA ratings, the LAATCL model is trained and tested.

4. Result and Discussion

In comparison to other models, the LAATCL model is assessed in this section. This experiment was conducted on a Windows 10 64-bit PC with an Intel® CoreTM i5-4210 CPU@2.80GHz, 8 GB of RAM, and a 1 TB hard drive. The Stanford Sentiment Treebank (SST) [21] and the CIFAR-10 [22] were the two datasets used. In all, there are 8,544 learning texts, 2,210 test texts, and 1,101 validation texts in the SST dataset. The CIFAR-10 dataset includes sixty thousand color photographs, with 6,000 images assigned to each of ten categories. Each image has a size of thirty-two by thirty-two pixels. The training phase made use of 50,000 photographs, whereas the testing phase made use of 10,000 images. For an objective comparison, Python 3.7.8 was used to run the existing models (Tree-RNN [12], Semi-SILDM [15], HT-CNN [18], and DTBDNN [20]) as well as the recently suggested LAATCL. The following is a definition of the performance evaluation measures:

Accuracy: Accuracy represents the proportion of correctly classified examples (both positive and negative) among all tested instances. It is computed using the formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

In this context, True Positive (TP) stands for the number of positively identified texts, True Negative (TN) for the number of negatively identified texts, False Positive (FP) for the number of negatively classified texts that were mistakenly classified as positive, and FN for the number of positively classified texts that were wrongly classified as negative.

Precision: Precision measures the accuracy of positive predictions and is calculated as:

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

Recall: Recall, sometimes-called sensitivity, measures how well the model can detect all positive events and is calculated as:

$$Recall = \frac{TP}{TN + FN} \quad (9)$$

F-measure: By averaging Precision and Recall, the F-measure strikes a good balance between the two parameters. Because of:

$$F - measure = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \qquad (10)$$

Figure 4 illustrates a comparative analysis of the LAATCL model with traditional models on the SST dataset. The LAATCL model shows enhanced precision by 7.03%, 5.43%, 2.91%, and 2.06% compared to DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN respectively. Additionally, the recall of the LAATCL is increased by 7.23%, 5.56%, 3.03%, and 2.16% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. The F-measure of the LAATCL is improved by 7.13%, 5.49%, 2.9%, and 2.11% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. Moreover, the accuracy of the LAATCL is enhanced by up to 7.24%, 5.6%, 3.05%, and 1.88% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. These findings indicate that the LAATCL model achieves superior classification performance on the SST dataset compared to the other models.

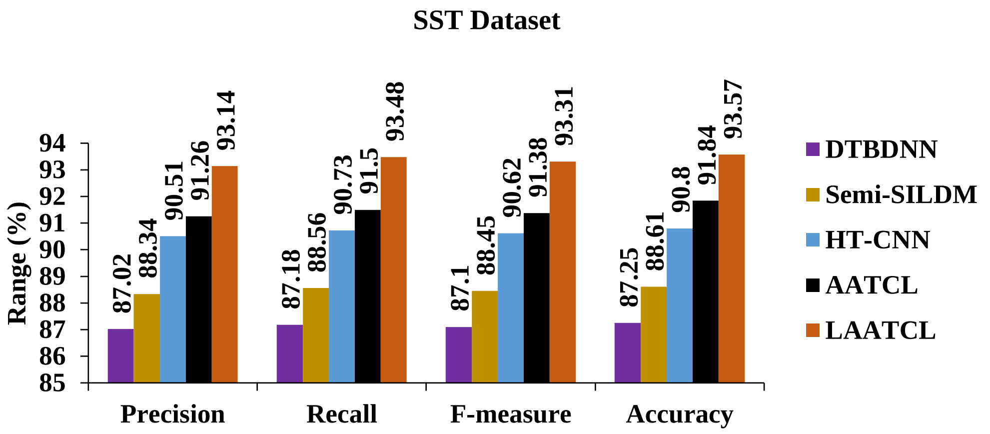

Figure 4. Comparison of LAATCL with traditional Tree-Based DL Classifiers Using SST Dataset

Figure 5 illustrates the comparative analysis of the LAATCL model with other traditional models evaluated on the CIFAR-10 dataset. The precision of the LAATCL has been enhanced by 5.5% and 4.42%, 2.99%, and 2.29% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. The recall of the LAATCL is enhanced by 5.66%, 4.41%, 3.14%, and 2.11% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. The F-measure of the LAATCL is improved by 5.58%, 4.42%, 3.06%, and 2.2% compared to the DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. In addition, the accuracy of the LAATCL is increased by up to 5.92%, 4.48%, 3.31%, and 2.27% compared to the DTBDNN, Semi-SILDM, HT-CNN and Tree-RNN models. It follows that, when tested on the CIFAR-10 dataset, the LAATCL model outperforms competing algorithms in terms of efficiency in classifying VA ratings of texts or images.

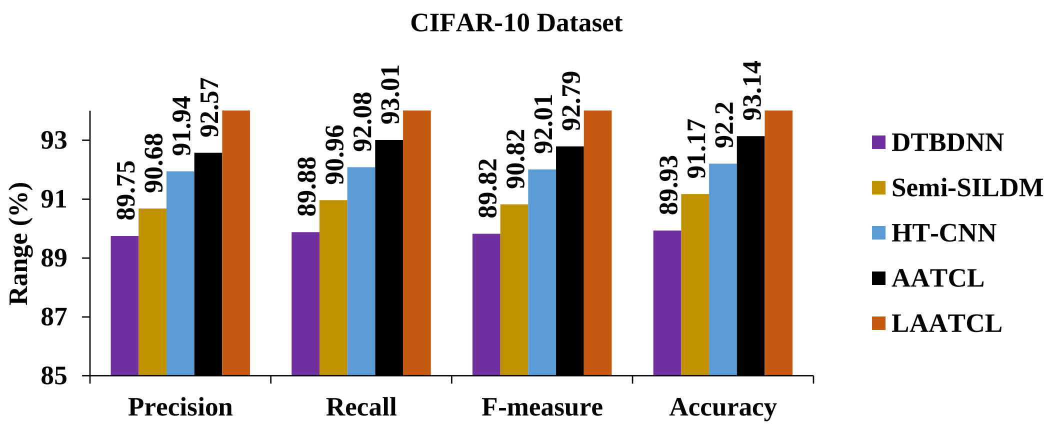

Figure 5. Comparison of LAATCL with traditional Tree-Based DL Classifiers Using CIFAR-10 Dataset

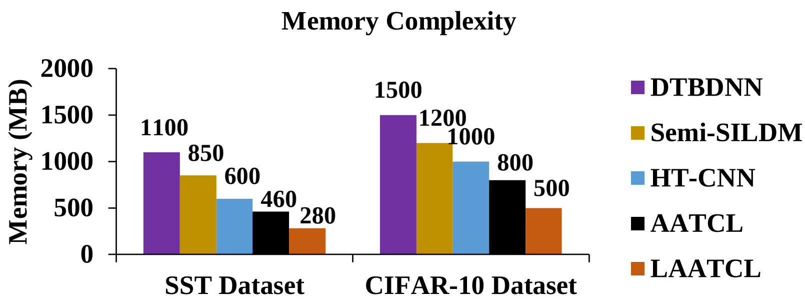

Figure 6. Memory Complexity for Different Tree-Based DL Classifiers

Figure 6 shows the memory complexity (in MB) of various tree-based DL classifiers on the SST and CIFAR-10 datasets. The LAATCL model significantly reduces memory complexity compared to other models. On the SST dataset, LAATCL reduces memory complexity by 74.55%, 67.06%, 53.33%, and 39.13% compared to DTBDNN, Semi-SILDM, HT-CNN, and Tree-RNN models. On the CIFAR-10 dataset, LAATCL reduces memory complexity by 66.67%, 58.33%, 50%, and 37.5% compared to the same models.

5. Conclusion

This study introduced the LAATCL model for DSA. It incorporated a clustering layer with the Tree-RNN to improve performance and reduce memory complexity by selecting reliable samples. This model utilized multi-convolution and a clustering layer with GSNMF for clustering highly correlated samples. GSNMF helped in identifying and selecting samples closest to the cluster centroid for input to the LSTM network, resulting in improved accuracy and reduced memory complexity. The findings demonstrated that the LAATCL model achieved 93.57% and 95.25% accuracy as well as 280kB memory complexity for SST and 500kB for CIFAR-10 datasets, outperforming state-of-the-art models.

Funding: This research received no external funding

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

References

[1] N. K. N. Neeraj and V. Maurya, "A review on machine learning (feature selection, classification and clustering) approaches of big data mining in different area of research," *J. Crit. Rev.*, vol. 7, no. 19, pp. 2610-2626, 2020.

[2] A. Goel, A. K. Goel, and A. Kumar, "The role of artificial neural network and machine learning in utilizing spatial information," *Spat. Inf. Res.*, vol. 31, no. 3, pp. 275-285, 2023.

[3] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, pp. 1-74, 2021.

[4] T. Yue et al., "Deep learning for genomics: From early neural nets to modern large language models," *Int. J. Mol. Sci.*, vol. 24, no. 21, p. 15858, 2023.

[5] E. Galván and P. Mooney, "Neuroevolution in deep neural networks: Current trends and future challenges," *IEEE Trans. Artif. Intell.*, vol. 2, no. 6, pp. 476-493, 2021.

[6] N. Rathi et al., "Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1-49, 2023.

[7] J. Bharadiya, "Convolutional neural networks for image classification," *Int. J. Innov. Sci. Res. Technol.*, vol. 8, no. 5, pp. 673-677, 2023.

[8] B. Kumar, O. Dikshit, A. Gupta, and M. K. Singh, "Feature extraction for hyperspectral image classification: A review," *Int. J. Remote Sens.*, vol. 41, no. 16, pp. 6248-6287, 2020.

[9] D. Roy, P. Panda, and K. Roy, "Tree-CNN: A hierarchical deep convolutional neural network for incremental learning," *Neural Netw.*, vol. 121, pp. 148-160, 2020.

[10] J. Wang, L. C. Yu, K. R. Lai, and X. Zhang, "Tree-structured regional CNN-LSTM model for dimensional sentiment analysis," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 581-591, 2019.

[11] H. He and Y. Fan, "A novel hybrid ensemble model based on tree-based method and deep learning method for default prediction," *Expert Syst. Appl.*, vol. 176, p. 114899, 2021.

[12] X. Ren, H. Gu, and W. Wei, "Tree-RNN: Tree structural recurrent neural network for network traffic classification," *Expert Syst. Appl.*, vol. 167, pp. 1-9, 2021.

[13] Z. Zang et al., "Tree-based ensemble deep learning model for spatiotemporal surface ozone (O3) prediction and interpretation," *Int. J. Appl. Earth Obs. Geoinf.*, vol. 103, p. 102516, 2021.

[14] O. Aouedi, K. Piamrat, and B. Parrein, "Ensemble-based deep learning model for network traffic classification," *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 4, pp. 4124-4135, 2022.

[15] F. Khozeimeh et al., "RF-CNN-F: Random forest with convolutional neural network features for coronary artery disease diagnosis based on cardiac magnetic resonance," *Sci. Rep.*, vol. 12, no. 1, p. 11178, 2022.

[16] S. C. C. Selvan et al., "Optimized hierarchical tree deep convolutional neural network of a tree-based workload prediction scheme for enhancing power efficiency in cloud computing," *Energies*, vol. 16, no. 6, p. 2900, 2023.

[17] Q. Cai, L. Niu, X. Shang, and H. Ding, "A self-supervised tree-structured framework for fine-grained classification," *Appl. Sci.*, vol. 13, no. 7, p. 4453, 2023.

[18] M. Arifuzzaman, M. R. Hasan, T. J. Toma, S. B. Hassan, and A. K. Paul, "An advanced decision tree-based deep neural network in nonlinear data classification," *Technologies*, vol. 11, no. 1, p. 24, 2023.

[19] "Stanford Sentiment Treebank," [Online]. Available: https://nlp.stanford.edu/sentiment/treebank.html.

[20] "CIFAR Dataset," [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.