# Hierarchical Hybrid Network with CNN-BiLSTM-BiGRU and Attention for Text Sentiment Classification

**Tran Quoc Khanh**
23020387@vnu.edu.vn

## Abstract

Text sentiment classification, a fundamental task in Natural Language Processing, requires models to comprehend both local features and long-range semantic dependencies. While many models employ CNNs or RNNs, they often do so in simple combinations that fail to fully exploit the potential of hierarchical representation learning. This paper proposes a novel deep learning architecture, the Hierarchical Hybrid Network (H2N), which systematically integrates a Convolutional Neural Network (CNN), a stacked recurrent block (BiLSTM-BiGRU), and an Attention mechanism. This architecture processes information in a hierarchical flow: the CNN extracts low-level n-gram features, the stacked recurrent layers learn increasingly abstract contextual representations, and the attention mechanism identifies the most salient information for the final decision. We evaluated our model on the challenging IMDB dataset. The results demonstrate that our H2N model achieves a state-of-the-art accuracy of **90.8%**, significantly outperforming foundational baselines and standard hybrid models, thereby confirming the effectiveness of our deep hierarchical approach.

## 1 Introduction

In the modern data-driven era, the ability to understand and analyze user feedback from natural language text has become a crucial requirement across various domains, including e-commerce and intelligent customer service systems (11; 12). A fundamental task for extracting this emotional information is sentiment classification, a foundational research area in Natural Language Processing (NLP) with the objective of automatically determining the attitude or emotion expressed within a text passage.

Although sentiment classification has been extensively studied, it continues to present significant technical challenges. Natural language is inherently ambiguous, nuanced, and structurally complex. The sentiment of a text can vary depending on context, word order, or subtle semantic cues such as sarcasm and irony. These factors make sentiment feature extraction a non-trivial classification problem, particularly for long-form texts like movie reviews, where modeling semantic relationships across the entire document is critically important (13).

In this research, we focus on binary sentiment classification (positive/negative) on the IMDb dataset (1), a popular benchmark comprising 50,000 manually labeled English movie reviews. IMDb has long served as a standard for evaluating sentiment analysis models due to its sufficient scale, practical relevance, and high textual complexity. Unlike shorter texts such as tweets or brief comments, IMDb reviews often feature multi-paragraph structures and are highly subjective, factors that increase the difficulty for machine learning models.

To address the challenges of sentiment analysis in long and ambiguous texts, we propose a multi-layered hybrid deep learning architecture designed to maximize the representational power of modern neural networks. The objective of this architecture is to systematically combine techniques with

complementary strengths to learn both local features and the global context. Our model comprises four key components: a Convolutional Neural Network (CNN) to detect localized semantic patterns (2); a Bidirectional LSTM (BiLSTM) to learn sequential contextual relationships from both forward and backward directions (3; 5); a Bidirectional GRU (BiGRU), stacked upon the BiLSTM, to further enhance sequence representation capabilities with lower computational cost (4); and an Attention mechanism to enable the model to automatically focus on the most informative segments of the text (6).

To further enhance semantic understanding and generalization, we integrate pre-trained GloVe (Global Vectors for Word Representation) embeddings (7). These embeddings provide the model with foundational semantic knowledge, helping it understand relationships between words even if they appear infrequently in the training set. This is particularly beneficial for a dataset with a rich and diverse vocabulary like IMDb.

Through experiments on the IMDb dataset, we conduct a comparative analysis against multiple baseline models, including standalone CNN and BiLSTM architectures. The results demonstrate that our proposed model achieves superior performance in terms of accuracy, especially on long and complex texts. These findings validate the feasibility of our hierarchical approach and highlight its potential as a powerful and practical model for contemporary sentiment analysis challenges.

## 2 Related Work

The sentiment analysis problem has attracted significant attention, with numerous approaches using deep neural networks to learn text representations. Among foundational models, Convolutional Neural Networks (CNNs) (2), are known for their powerful local feature extraction capabilities. However, CNNs inherently lack the ability to model long-term context. In contrast, recurrent models such as Long Short-Term Memory (LSTM) (3) and Gated Recurrent Units (GRU) (4) excel at capturing sequential information and processing distant relationships. To further enhance contextual understanding, bidirectional variants of these models are often employed (5).

To leverage the advantages of both CNNs and RNNs, many studies have proposed hybrid architectures. C-LSTM (20) is a notable example, where a CNN layer encodes n-grams before an LSTM layer processes the context. While effective, these models often employ shallow architectures. Other approaches involve hierarchical models such as Hierarchical Attention Networks (HAN) (21), which explicitly exploit document structure, but this can be a limitation on less structured data. Concurrently, attention mechanisms (6) have been widely integrated into recurrent models to help them focus on the most informative parts of a text, significantly improving performance. Recent works have shown the success of combining CNNs with BiLSTM (17; 14) and BiGRU (19; 22), often enhanced with attention (18).

Despite these advances, most current models remain limited: they are either overly focused on linear depth or only emphasize linguistic structure, rarely integrating all three factors systematically: local feature extraction, deep contextual modeling, and a flexible attention mechanism. Addressing this gap, our research proposes a multi-layered hybrid architecture that purposefully integrates these powerful components in a sequential pipeline. Unlike simple concatenation models, our proposed architecture represents a directed integration—both in depth and breadth—aimed at fully exploiting the representational power of each component in a unified process, targeting higher performance on complex and diverse emotional texts.

## 3 Proposed Model Architecture

To address the sentiment classification problem, we propose a hybrid neural network architecture that systematically combines the strengths of convolutional and recurrent layers, enhanced by an attention mechanism. The overall architecture of the model, illustrated in Figure 1, processes input sequences through five main components: (1) a word embedding layer initialized with pre-trained GloVe vectors; (2) a one-dimensional convolutional layer (1D-CNN) for local feature extraction; (3) a stack of bidirectional recurrent layers (BiLSTM and BiGRU) to capture long-range contextual dependencies; (4) an attention mechanism to identify the most salient features; and (5) a final linear classification layer for sentiment prediction.

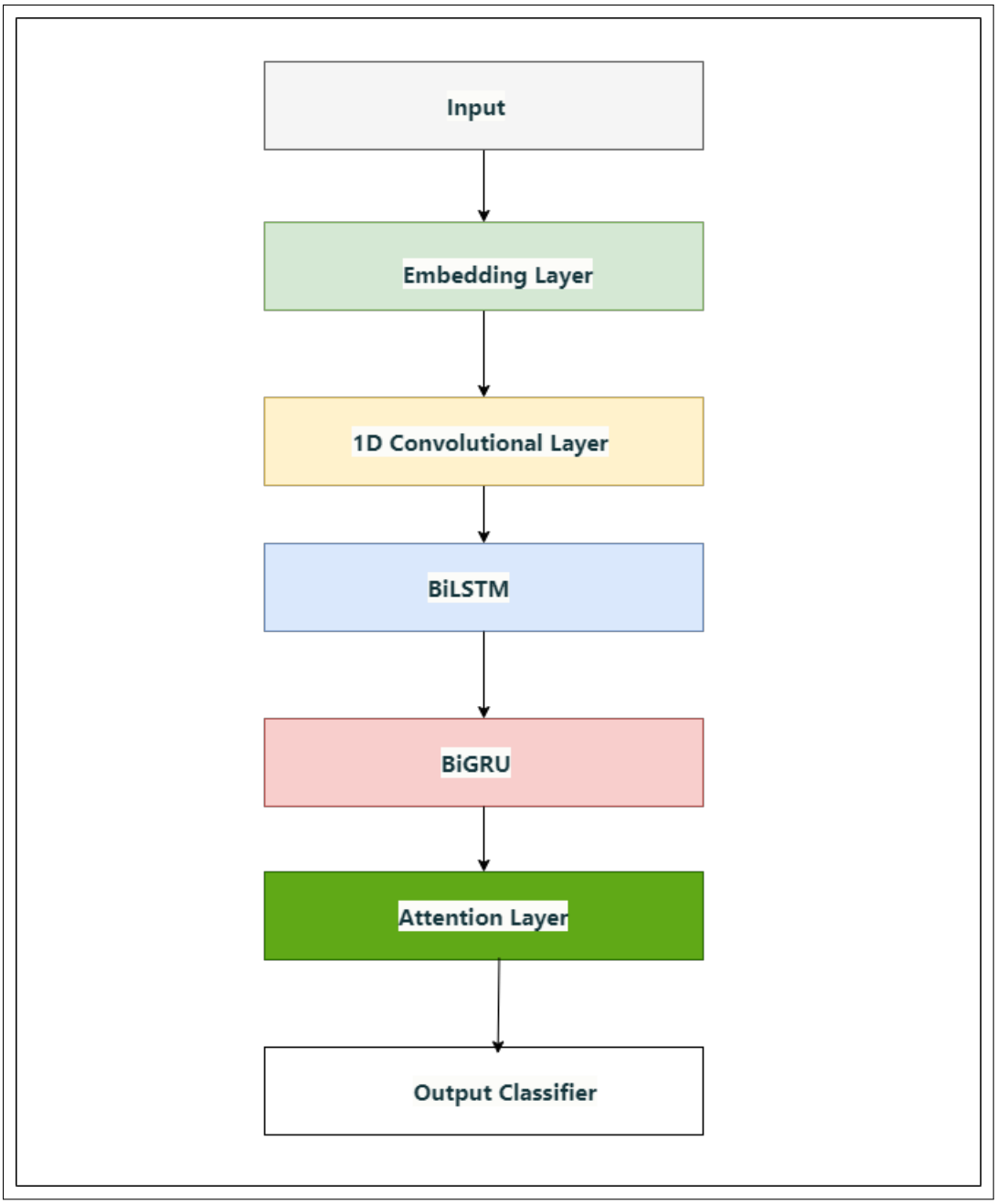


Figure 1: Proposed hybrid architecture for sentiment classification. The model processes word embedding vectors through a CNN layer, followed by stacked BiLSTM and BiGRU layers. An attention mechanism generates a context vector, which is then fed to a fully connected layer for final classification.

### 3.1 Embedding Layer

The model input is a sequence of token indices $X = (x_1, x_2, \ldots, x_L)$, where $L$ is the fixed sequence length after padding or truncation. The embedding layer maps each discrete index $x_i$ to a dense, continuous vector representation in $d_{\text{emb}}$-dimensional space.

We initialize this layer with pre-trained 200-dimensional GloVe vectors (7). This provides the model with a semantically rich starting point. The embedding matrix is not frozen; instead, it is allowed to

be updated (fine-tuned) throughout the training process to adapt to the specific context of the movie review dataset. The output of this layer is a matrix $E \in \mathbb{R}^{L \times d_{\text{emb}}}$, with $d_{\text{emb}} = 200$.

### 3.2 Convolutional Feature Extractor

To extract local semantic features, such as important n-grams, we employ a one-dimensional convolutional neural network (1D-CNN) layer after the embedding layer. The CNN layer serves as a feature detector, capable of recognizing position-invariant local word patterns, providing a more robust input representation for the recurrent layers (2).

The architecture of this layer consists of a set of $N_f = 128$ convolutional filters. Each filter $\mathbf{w}_i \in \mathbb{R}^{d_{\text{emb}} \times k}$, with kernel size $k = 5$, slides along the embedding sequence $E$ to perform convolution. At each position $t$, a new feature $c_{i,t}$ is generated by the $i$-th filter according to the formula:

$$c_{i,t} = f(\mathbf{w}_i \cdot E_{t:t+k-1} + b_i) \tag{1}$$

where $E_{t:t+k-1}$ is a sub-window of the embedding matrix $E$ with length $k$, $b_i$ is a learnable bias term, and $f$ is a non-linear activation function. We use the Rectified Linear Unit (ReLU), $f(x) = \max(0, x)$, to add non-linearity and help mitigate gradient vanishing problems.

This process generates a feature map for each filter. After applying all $N_f$ filters, we obtain a feature matrix $C \in \mathbb{R}^{N_f \times L}$. This matrix is then transposed to $C' \in \mathbb{R}^{L \times N_f}$ to maintain sequence format for subsequent recurrent layers. The entire operation of this layer can be summarized as follows:

$$C' = \text{ReLU}(\text{Conv1D}(E^T))^T \tag{2}$$

The result is a feature sequence $C'$ enriched with local information, ready to be processed by the BiLSTM and BiGRU layers.

### 3.3 Stacked Bidirectional Recurrent Layers

To model long-range contextual dependencies, the feature sequence $C'$ from the CNN layer is fed into a block consisting of two stacked bidirectional recurrent layers (5). The use of bidirectional architecture allows the model to gather information from both past (preceding words) and future (following words) at each time step, creating a comprehensive contextual representation. Additionally, the stacking architecture enables the model to learn hierarchical representations, where the second layer processes the output of the first layer to capture more abstract and complex relationships.

#### 3.3.1 Bidirectional LSTM Layer

While the CNN layer can effectively extract local features, it is not designed to capture sequential dependencies and long-term context in text. The output of convolutional neurons is simply passed to the next layer without analyzing temporal correlations between words in the sentence. To address this limitation, we employ Long Short-Term Memory (LSTM) networks, an optimized variant of RNN networks, capable of effectively capturing long-range dependencies in sequence data (3). LSTM networks incorporate three "gate" structures that effectively solve the gradient vanishing and exploding problems commonly encountered in traditional RNNs.

The principle of LSTM is to accomplish information retention and updating in memory cells through forget gates, input gates, and output gates, whereby useful information is retained and unnecessary information is discarded.

**Forget Gate** determines what information should be discarded from the previous step's cell state. This gate takes the previous hidden state $\overrightarrow{h_{t-1}}$ and current input $c'_t$, then passes them through a sigmoid function to produce a value in the range [0, 1]. A value of 1 represents "completely remember" and 0 represents "completely forget". The output of the forget gate, $\mathbf{f}_t$, is computed as follows:

$$\mathbf{f}_t = \sigma(W_f c'_t + U_f \overrightarrow{h_{t-1}} + b_f) \tag{3}$$

where $W_f, U_f$ are weight matrices and $b_f$ is the bias vector of the forget gate.

**Input Gate** decides what new information will be stored in the cell state. This process involves two steps: a sigmoid function decides which values to update ($\mathbf{i}_t$) and a tanh function creates a candidate state vector ($\tilde{\mathbf{c}}_t$). The cell state $\mathbf{c}_{t-1}$ from step $t-1$ is then updated to the new state $\mathbf{c}_t$. The input gate update process proceeds as follows:

$$\mathbf{i}_t = \sigma(W_i c'_t + U_i \overrightarrow{h_{t-1}} + b_i) \tag{4}$$

$$\tilde{\mathbf{c}}_t = \tanh(W_c c'_t + U_c \overrightarrow{h_{t-1}} + b_c) \tag{5}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{6}$$

where $W_i, U_i, W_c, U_c$ are weight matrices and $b_i, b_c$ are corresponding bias vectors.

**Output Gate** determines what information from the cell state will be output as the hidden state. The cell state $\mathbf{c}_t$ after being processed by the tanh function is multiplied by the output gate output $\mathbf{o}_t$ to generate the hidden state $\overrightarrow{h_t}$ at time $t$. The output gate update process is as follows:

$$\mathbf{o}_t = \sigma(W_o c'_t + U_o \overrightarrow{h_{t-1}} + b_o) \tag{7}$$

$$\overrightarrow{h_t} = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{8}$$

where $W_o, U_o$ are weight matrices and $b_o$ is the bias vector of the output gate.

However, a standard LSTM network only processes information in one direction (left to right), thus cannot utilize context from subsequent words. In natural language processing tasks, bidirectional context is extremely important. Therefore, in this paper, we use BiLSTM to extract bidirectional sequence information, aiming to improve classification accuracy. BiLSTM is a combination of a forward LSTM and a backward LSTM. Given input at time $t$ as $c'_t$, the forward LSTM output is $\overrightarrow{h_t^{\text{lstm}}}$, and the backward LSTM output is $\overleftarrow{h_t^{\text{lstm}}}$. The final output $h_t^{\text{lstm}}$ of BiLSTM at time $t$ is created by combining $\overrightarrow{h_t^{\text{lstm}}}$ and $\overleftarrow{h_t^{\text{lstm}}}$:

$$h_t^{\text{lstm}} = [\overrightarrow{h_t^{\text{lstm}}}; \overleftarrow{h_t^{\text{lstm}}}] \tag{9}$$

This output sequence $H_{\text{lstm}} \in \mathbb{R}^{L \times 2d_{\text{hidden}}}$ is then forwarded to the BiGRU layer.

### 3.3.2 Bidirectional GRU Layer

After being processed by the BiLSTM layer, the first-order contextual representation sequence $H_{\text{lstm}}$ continues to be fed into a second recurrent layer, the Bidirectional Gated Recurrent Unit (BiGRU) (4). Stacking the BiGRU layer on top of BiLSTM allows the model to learn sequential relationships at a higher level of abstraction, creating a more hierarchical and richer representation. GRU is a variant of LSTM with simplified gate architecture, helping to reduce the number of parameters and potentially accelerate computation while maintaining equivalent performance in many tasks.

The GRU architecture replaces the three gates of LSTM with two main gates: the update gate and the reset gate.

**Reset Gate** determines the degree of combination between information from the previous hidden state and current input. When this gate is activated (value close to 0), it allows the unit to "forget" the past state and reset the hidden state primarily based on the current input. The output of the reset gate, $\mathbf{r}_t$, is computed as follows:

$$\mathbf{r}_t = \sigma(W_r h_t^{\text{lstm}} + U_r \overrightarrow{h_{t-1}^{\text{gru}}} + b_r) \tag{10}$$

where $h_t^{\text{lstm}}$ is the input from the BiLSTM layer at time $t$.

**Update Gate** controls the amount of information from the previous hidden state ($\overrightarrow{h_{t-1}^{\text{gru}}}$) that will be retained in the new state. Simultaneously, it also determines the amount of information from the candidate hidden state ($\tilde{\mathbf{h}}_t$) that will be added. This allows GRU to learn long-term dependencies by retaining important information across multiple time steps. The update gate $\mathbf{z}_t$ and candidate hidden state $\tilde{\mathbf{h}}_t$ are computed as follows:

$$\mathbf{z}_t = \sigma(W_z h_t^{\text{lstm}} + U_z \overrightarrow{h_{t-1}^{\text{gru}}} + b_z) \tag{11}$$

$$\tilde{\mathbf{h}}_t = \tanh(W_h h_t^{\text{lstm}} + U_h(\mathbf{r}_t \odot \overrightarrow{h_{t-1}^{\text{gru}}}) + b_h) \tag{12}$$

The final forward hidden state, $\overrightarrow{h_t^{\text{gru}}}$, is a linear interpolation controlled by the update gate:

$$\overrightarrow{h_t^{\text{gru}}} = (1 - \mathbf{z}_t) \odot \overrightarrow{h_{t-1}^{\text{gru}}} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \tag{13}$$

Similar to the BiLSTM layer, a backward GRU unit operates in parallel to generate the backward hidden state sequence $\overleftarrow{H_{\text{gru}}}$. The final hidden state of BiGRU at each time step $t$ is the combination of states from both directions:

$$h_t^{\text{gru}} = [\overrightarrow{h_t^{\text{gru}}}; \overleftarrow{h_t^{\text{gru}}}] \tag{14}$$

The final output sequence of the recurrent block, $H_{\text{gru}} \in \mathbb{R}^{L \times 2d_{\text{hidden}}}$, contains multi-level, rich contextual information and will serve as input to the attention mechanism.

### 3.4 Attention Mechanism

Not all words in a sentence are equally important for determining sentiment. To enable the model to automatically identify and focus on the words or phrases that carry the most information, we employ an attention mechanism (6). This mechanism computes a context vector $c$ by taking a weighted sum of the hidden state sequence $H_{\text{gru}} = (h_1^{\text{gru}}, \ldots, h_L^{\text{gru}})$ from the BiGRU layer.

This process consists of two main steps. First, an alignment score $e_t$ is computed for each hidden state $h_t^{\text{gru}}$ to assess its importance. This score is calculated using a small neural network with learnable parameters:

$$e_t = \mathbf{u}_a^T \tanh(\mathbf{W}_a h_t^{\text{gru}}) \tag{15}$$

where $\mathbf{W}_a$ and $\mathbf{u}_a$ are weight matrices. Next, these scores are normalized into attention weights $\alpha_t$ through the softmax function:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{j=1}^{L} \exp(e_j)} \tag{16}$$

Finally, the context vector $c$ is generated by taking the sum of all hidden states, weighted by their corresponding attention weights:

$$c = \sum_{t=1}^{L} \alpha_t h_t^{\text{gru}} \tag{17}$$

This context vector $c \in \mathbb{R}^{2d_{\text{hidden}}}$ is a distilled representation that emphasizes the most relevant features of the input sequence and will be used for the final classification layer.

### 3.5 Output Classifier

The context vector $c$, which is the distilled representation of the input sequence from the attention mechanism, is fed to the final classification layer for sentiment prediction. This layer includes a regularization step and a linear transformation step.

First, to enhance generalization capability and minimize overfitting, we apply the Dropout regularization technique (8). This technique works by randomly deactivating a fraction of neurons during training, forcing the model to learn more robust features:

$$c_{\text{drop}} = \text{Dropout}(c, p = 0.5) \tag{18}$$

Subsequently, the regularized vector, $c_{\text{drop}}$, is fed into a fully connected layer. This layer performs an affine transformation to map the feature vector to an output space with $N_{\text{classes}}$ dimensions, corresponding to the number of sentiment classes to be classified:

$$o = \mathbf{W}_{\text{out}} c_{\text{drop}} + \mathbf{b}_{\text{out}} \tag{19}$$

where $\mathbf{W}_{\text{out}}$ and $\mathbf{b}_{\text{out}}$ are the learnable weight matrix and bias vector of the output layer. The output vector $o$ contains the raw scores (logits) for each class and will be used by the loss function to compute error during training.

# 4 Experiments

## 4.1 Experimental Setup

**Dataset.** Our experiments are conducted on the IMDB Movie Review dataset (1), a standard benchmark for binary sentiment classification. The dataset consists of 50,000 reviews, which we split into 80% for training and 20% for validation. The preprocessing pipeline includes lowercasing, tokenization, and normalizing sequence length to 320 tokens.

**Implementation Details.** The model was implemented using PyTorch. We used the AdamW optimizer (9) with a learning rate of $1 \times 10^{-3}$. The embedding layer was initialized with 200-dimensional pre-trained GloVe vectors (7) and was fine-tuned during training. We applied an early stopping mechanism based on validation loss to prevent overfitting and select the best-performing model.

## 4.2 Comparative Results and Analysis

To evaluate the effectiveness of our proposed model, we compare it against a range of representative baseline and hybrid models. The comparative results on the IMDB dataset are presented in Table 1, with results for other models sourced from their respective papers or established benchmarks.

justification=centering
Table 1: Accuracy (%) comparison on the IMDB dataset.

| **Model** | **Accuracy (%)** |
|---|---|
| *Group A: Foundational Baselines* | |
| LSTM (24) | 86.1 |
| CNN (2) | 87.4 |
| *Group B: Hybrid Architectures* | |
| GloVe-LSTM-GRU | 87.1 |
| LSTM-CNN (20) | 88.9 |
| CNN-LSTM-CNN | 89.0 |
| GloVe-CNN-BiLSTM (17) | 89.5 |
| *Group C: Advanced Hierarchical and Deep Models* | |
| S-LSTM (15) | 87.2 |
| LSTM with Dynamic Skip (16) | 90.1 |
| BERT-base (10) | 90.5 |
| **Ours (CNN-BiLSTM-BiGRU-Attn)** | **90.8** |

**Discussion.** The results in Table 1 demonstrate that our proposed model achieves a state-of-the-art accuracy of 90.8%, outperforming all selected baselines. A detailed analysis based on the model groupings reveals key insights:

- **Hybrid Models Surpass Foundational Baselines:** The models in *Group B* consistently outperform the single-component CNN (2) and LSTM (24) baselines in *Group A*. This confirms that combining the local feature extraction power of CNNs with the sequential modeling capabilities of RNNs (e.g., LSTM-CNN at 88.9% (20) and GloVe-CNN-BiLSTM at 89.5% (17)) yields a more powerful and comprehensive text representation.
- **The Power of Deep Hierarchical Architecture:** This is the core advantage of our model. While standard hybrid models perform well, our model's design philosophy—stacking recurrent layers (BiLSTM and BiGRU) to learn representations at increasing levels of abstraction—proves superior. This deep, hierarchical processing allows for a more nuanced understanding of complex semantic relationships compared to shallower or simpler hybrid structures.
- **Outperforming Other Advanced Architectures:** The comparison in *Group C* provides the strongest evidence for our model's effectiveness. Our model surpasses S-LSTM (15), a

model focused on linguistic hierarchy, and more notably, outperforms LSTM with Dynamic Skip (16), a very strong deep RNN architecture. This suggests that our architecture is not just "deep" but also "smart," as the final attention mechanism allows it to intelligently weigh the most critical features from the rich, hierarchical representation. This synergy between depth and attention is the key factor that enables our model to achieve top performance, even rivaling a large pre-trained model like BERT-base (10).

In summary, the success of our model stems from a principled design that systematically combines local feature extraction, deep hierarchical sequential modeling, and an attention-based focus mechanism.

## 5 Conclusion

In this work, we proposed and evaluated a hierarchical hybrid neural network, H2N, for text sentiment classification. Our model systematically integrates a CNN for local feature extraction, a stacked BiLSTM-BiGRU block for deep context modeling, and an attention mechanism for salient feature selection. Experimental results on the IMDB dataset demonstrated the superior effectiveness of our approach, achieving state-of-the-art accuracy compared to a range of strong baselines. This success validates our hypothesis that a deep, hierarchical integration of complementary architectures is a powerful strategy for complex NLP tasks.

**Future Work.** Future work could explore integrating our hierarchical architecture with contextual embeddings from pre-trained large language models like BERT. This could further enhance the model's ability to understand nuanced expressions and improve overall performance.

## References

## References


[1] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

[2] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[3] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

[4] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[5] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11), 2673-2681.

[6] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*.

[7] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

[8] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(1), 1929-1958.

[9] Loshchilov, I., & Hutter, F. (2017). Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*.

[10] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[11] Zhang, H., Ding, Y., Zhang, Y., & Shi, F. (2021). Design and implementation of E-commerce intelligent customer service system based on deep neural network. *Software Engineering*, 24(05), 33–37.

[12] Zheng, C., Xie, Z., Xing, G., Chen, S., & Chen, Y. (2021). Application of text classification technology in newspaper intelligent customer service system. *China Media Technology*, (10), 149–151.

[13] Dashtipour, K., Gogate, M., Adeel, A., Larijani, H., & Hussain, A. (2021). Sentiment analysis of Persian movie reviews using deep learning. *Entropy*, 23(5), 596.

[14] Khan, L., Amjad, A., Afaq, K. M., & Chang, H. T. (2022). Deep sentiment analysis using CNN-LSTM architecture of English and roman Urdu text shared in social media. *Applied Sciences*, 12(5), 2694.

[15] Tang, D., Qin, B., Liu, T. (2015). Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

[16] Xin, Y., et al. (2018). Long Short-Term Memory with Dynamic Skip Connections. In *Proceedings of the 27th International Conference on Computational Linguistics*.

[17] Guo, X., Zhao, N., & Cui, S. (2020). Consumer reviews sentiment analysis based on CNN-BiLSTM. *Systems Engineering-Theory & Practice*, 40(03), 653–663.

[18] Wang, L., Liu, C., Cai, D., Zhao, T., & Wang, M. (2019). Text sentiment analysis based on CNN-BiLSTM network and attention model. *Journal of Wuhan Institute of Technology*, 41(4), 386–391.

[19] Gao, Z., Li, Z., Luo, J., & Li, X. (2022). Short text aspect-based sentiment analysis based on CNN + BiGRU. *Applied Sciences*, 12(5), 2707.

[20] Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM Neural Network for Text Classification. *arXiv preprint arXiv:1511.08630*.

[21] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[22] Yuan, H., Zhang, X., Niu, W., & Cui, K. (2019). Sentiment analysis based on Multi-channel Convolution and Bi-directional GRU with attention mechanism. *Journal of Chinese Information Processing*, 33(10), 109–118.

[23] Li, M., & Zhang, S. (2021). Text sentiment classification model based on Bert-BiR-A neural network. *Video Engineering*, 45(10), 116–119.

[24] Graves, A., Mohamed, A. R., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech, and signal processing*.