Group Assignment - Group 45

Natural Language Processing - SC4002

Date: November 12, 2025

Contributions

Overview of the individual contributions of each team member.

Student	**E-mail**	**Contributions**
Alexander Jonsson	N2503534A@e.ntu.edu.sg	Laid the groundwork for the project by setting up the original repository and designing its structure. Wrote the initial source code for tasks 01, 02 and 03a&b, ensuring that the main logic and results for the respective tasks were implemented and in place.
Petter Ollinen	N2503542E@e.ntu.edu.sg	Contributed by aligning existing and newly developed plots and visualizations for consistency, and continuously verified the results for accuracy.
Brianna Fan	N2503751K@e.ntu.edu.sg	Implemented part 3.3 (further improvement of the model). Organized Overleaf structure and worked on section 4 of the report.
Samuel Söderberg	N2504504L@e.ntu.edu.sg	Corrected minor problems with Alexander's initial solution drafts for parts 1 and 2, expanded README with report draft for parts. Finalised mitigation strategy implementations for OOV words and its evaluation together with Alexander (SIF-weighted, simple mean, zero vector) + other minor corrections.
Mateusz Hibner	N2504055K@e.ntu.edu.sg	Contributed by writing sections 1–3 of the report and assisting in resolving minor inaccuracies.
Chi Seo Hyeon	SEOHYEON001@e.ntu.edu.sg	Implemented part 3.4 (Improvement strategies for weak categories) and worked on section 4 of the report.

Github link: https://github.com/JonssonAlexander/NLP_Project_group45

1 Introduction

Topic classification in the field of NLP (Natural Language Processing) is the task of assigning predefined categories or labels to the text based on its content. Its usefulness lies in enabling the automatic understanding and categorization of large volumes of unstructured text, allowing faster information retrieval, better organization, and meaningful analysis. This feature enables a number of applications, including spam screening and news classification.

Word embeddings are numerical vector representations of words that encapsulate their meaning and relationships with one another. Once words are converted into embeddings, those that appear in similar contexts are positioned closer together in the continuous vector space, reflecting their semantic relatedness. Conversely, embeddings that are far apart indicate words with distinct or unrelated meanings.

The learning objectives of this assignment were to gain hands-on experience with real-world NLP applications by applying the theoretical concepts covered during the semester. Precisely, the project focused on building and evaluating models for sentence-level topic classification using pretrained word embeddings and neural architectures such as recurrent neural network (RNN), Bidirectional LSTM (BiLSTM), Bidirectional Gated Recurrent Unit (BiGRU), and convolutional neural network (CNN). Throughout the process we strived to understand how word representations influence model performance, how to handle problems like out-of-vocabulary (OOV) words, how to visualize and analyze embeddings, and discover how different architectures perform on this specific task.

2 Preparing Word Embeddings

2.1 Dataset

The dataset used for this project is the TREC Question Classification dataset, which contains short, factual questions which are grouped into six coarse-grained categories:

- ABBR - abbreviation (e.g., *What does the abbreviation AIDS stand for?*)
- DESC - description (e.g., *How did serfdom develop in and then leave Russia?*)
- ENTY - entity (e.g., *What fowl grabs the spotlight after the Chinese Year of the Monkey?*)
- HUM - human (e.g., *Who was the inventor of silly putty?*)
- LOC - location (e.g., *What sprawling U.S. state boasts the most airports?*)
- NUM - numeric value (e.g., *How many Community Chest cards are there in Monopoly?*)

For this project, we used the "Training set 5(5500 labeled questions)" version which contains 5,500 samples. This set was divided into training (80%) and validation (20%). "Test set: TREC 10 questions" containing 10 samples was used for testing. A fixed random seed (42) was used during the split to ensure reproducibility.

2.2 Vocabulary Size

Text preprocessing was performed to standardize the input format and prepare the data for embedding and model training. All the questions were tokenized using the spaCy English model (en_core_web_sm), while converting them to lowercase in order to reduce vocabulary sparsity. A minimum frequency threshold of 1 was used during vocabulary construction so that every token appearing at least once in the training corpus was included. Two special tokens were added: <pad> for sequence padding, and <unk> for representing words not present in the vocabulary.

After preprocessing, the resulting training corpus contained 4,361 documents and 45,591 tokens, forming a vocabulary of 7,479 tokens (including the special symbols <pad> and <unk>)

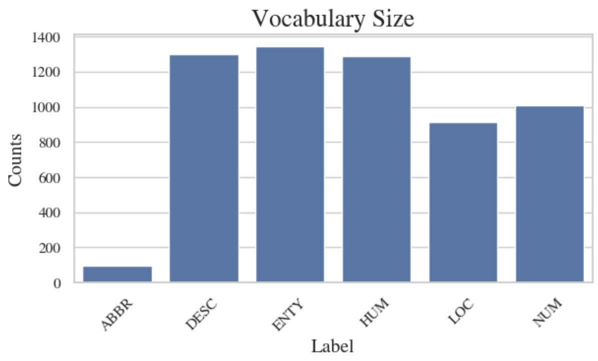

Figure 1: Histogram of the categories in the vocabulary.

2.3 Out-of-Vocabulary (OOV) Analysis

After mapping each of the 7,479 tokens to the GloVe 6B-100d pretrained embeddings, we found 197 unique out-of-vocabulary (OOV) words - tokens that appeared in the training data but were not present in GloVe's vocabulary. That corresponds to a total of 215 OOV token occurrences across the training set.

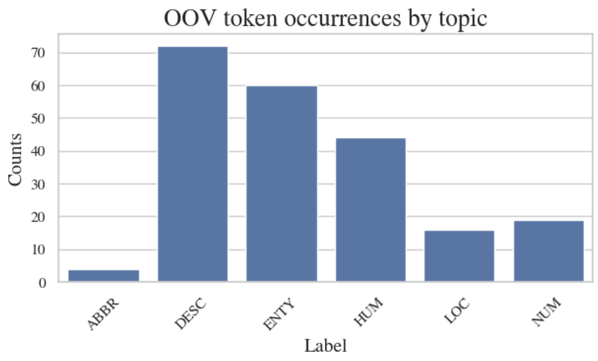

Figure 2: Histogram of the categories of the OOV words.

The highest OOV counts appear in DESC, ENTY, and HUM classes. We suspect the reason to be domain-specific terminology and proper nouns that are underrepresented in the general-domain GloVe corpus.

2.4 Out-of-Vocabulary (OOV) Mitigation Strategy

In order to resolve the problem of OOV we used global mean vector initialisation. First, the global mean of all in-vocabulary embedding vectors was computed. Special tokens (<pad> and <unk>) and OOV tokens were left out since they do not carry semantic meaning and would otherwise bias the mean vector. Subsequently, the mean vector was assigned to all OOV tokens as their initial representation, while keeping the embeddings trainable so the model can adapt OOV vectors during training.

This approach provides a more meaningful initialisation compared to e.g., random noise. It positions the OOV tokens near the semantic "centroid" of the vocabulary, and allows the model

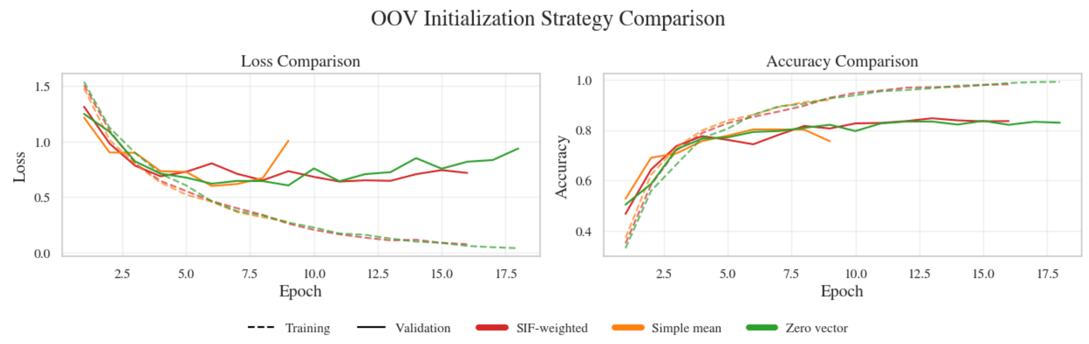

Figure 3: Training loss and accuracy curves for different OOV initialization strategies.

to fine-tune these vectors based on task-specific context during training. On top of that, the process is computationally efficient and simple to implement.

As a result all the 197 OOV tokens plus 2 special characters were mitigated, resulting in 199 initialised vectors.

2.5 Semantic Clustering Across Topics

In order to get an idea of the resulting embedding mapping we selected the 20 most frequent words and their embeddings from each topic category in the training set and projected them into 2D space using Principal Component Analysis (PCA).

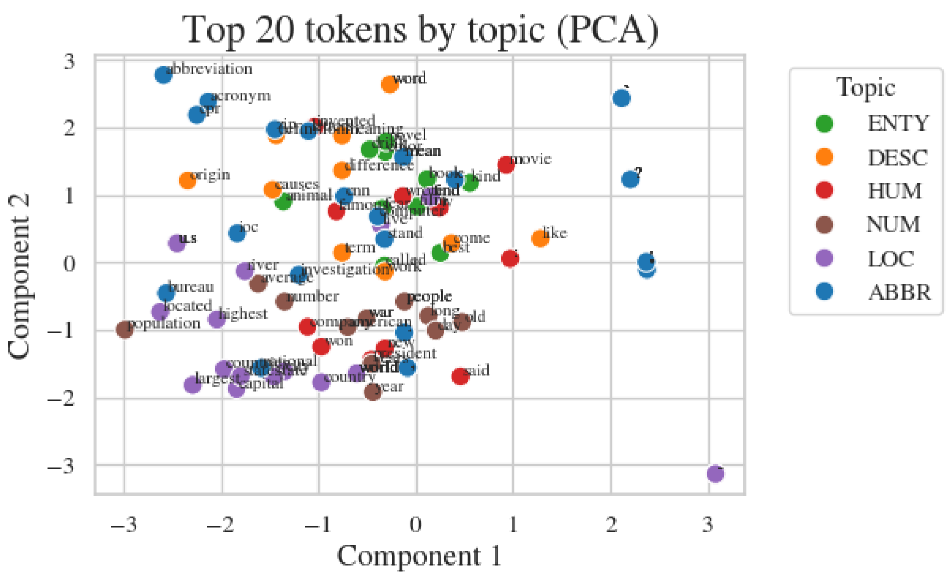

Figure 4: PCA plot of the 20 most frequent word of each topic.

The PCA graph shows several clusters. The LOC category tokens (purple) are grouped closely together, suggesting that the location-related words share strong semantic similarity. Similarly, NUM tokens (brown) form a tightly packed region, reflecting consistent numeric or quantitative semantics, and ENTY tokens (green) follow a similar clustering pattern. These concentrated areas suggest high semantic coherence within these topics.

In contrast, tokens from the DESC, HUM, and ABBR categories are more widely spread throughout the embedding space. This dispersion reflects the broader and more diverse semantic nature of these question types, where vocabularies are not limited to a single semantic scope.

In addition, punctuation marks such as the question mark "?", belonging to the ABBR class, appear as isolated points, due to their unique syntactic role and lack of direct semantic.

3 RNN Model Training & Evaluation

3.1 Best RNN Configuration

The best model was trained for 40 epochs with a batch size of 64, using the Adam optimizer and a learning rate of 0.001. The network includes a hidden dimension of 128 with 1 layer, dropout of 0.1, weight decay of 0.0, and gradient clipping at 1.0. Max pooling was used, and the best performance was achieved at epoch 15 with a validation accuracy of 0.853 and a test accuracy of 0.800.

3.2 Regularisation Strategies

We tested several regularization techniques to prevent overfitting, including different dropout rates ranging from 0.0 to 0.6, different L2 weight decay values ranging from 0.0 to 1×10^{-2}, gradient clipping, and early stopping. In total, 49 configurations were tested. The best combination was dropout = 0.1, weight decay = 0.0, grad_clip = 1.0 with val_accuracy = 0.841.

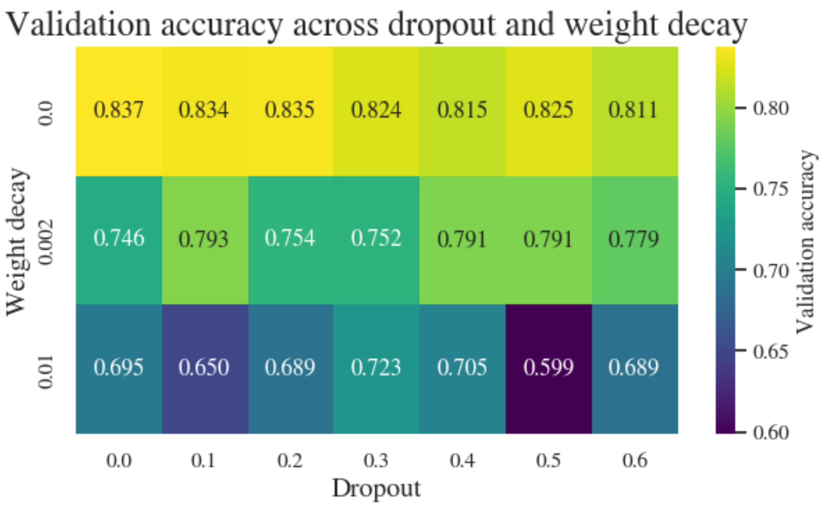

Figure 5: Comparison of tested regularization strategies.

3.3 Training Dynamics

During the first 10 epochs, the model exhibited a steady decrease in loss and a corresponding improvement in accuracy. After this point, the training process plateaued, indicating convergence. Early stopping was applied to prevent overfitting.

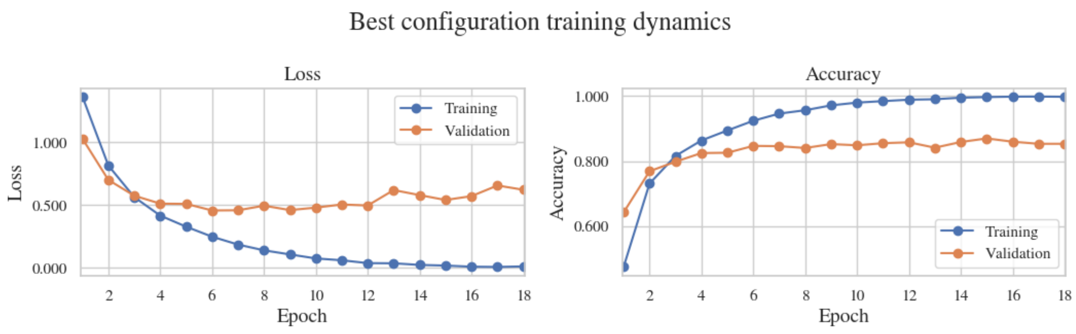

Figure 6: Best configuration RNN training dynamics.

3.4 Sentence Aggregation Methods

Different pooling strategies were explored to derive a final sentence representation. Max pooling achieved the highest test accuracy (88.2%), followed by mean pooling (87.4%). The last hidden state and attention pooling methods performed slightly worse, with accuracies of 86.4% and 86.2% respectively.

Table 1: Test accuracy of different pooling strategies for sentence representation.

Pooling Method	Test Accuracy (%)
Max pooling	88.2
Mean pooling	87.4
Last hidden state	86.4
Attention pooling	86.2

3.5 Topic-wise Accuracy

The model achieved the highest accuracy on the DESC (description) and LOC (location) categories, with 97% and 93% respectively. These categories generally include more descriptive or geographic terms that appear frequently in the training data, allowing the model to learn reliable patterns. Similarly, HUM (human) and NUM (numeric) questions performed well (0.89 and 0.88). We suspect the reason to be the distinctive lexical cues such as *who*, *how many*, or *when*, which are unique to these categories. In contrast, ENTY (entity) and ABBR (abbreviation) categories showed notably lower accuracy (0.63 and 0.78). These classes demonstrate higher lexical diversity and contain more rare or context-dependent phrases (e.g., specific names, acronyms) that may not be well represented or present in the GloVe embeddings. The small sample size for ABBR questions also contributes to weaker generalization. Overall, the topic-wise variation reflects the fact that consistent wording patterns support accurate classification.

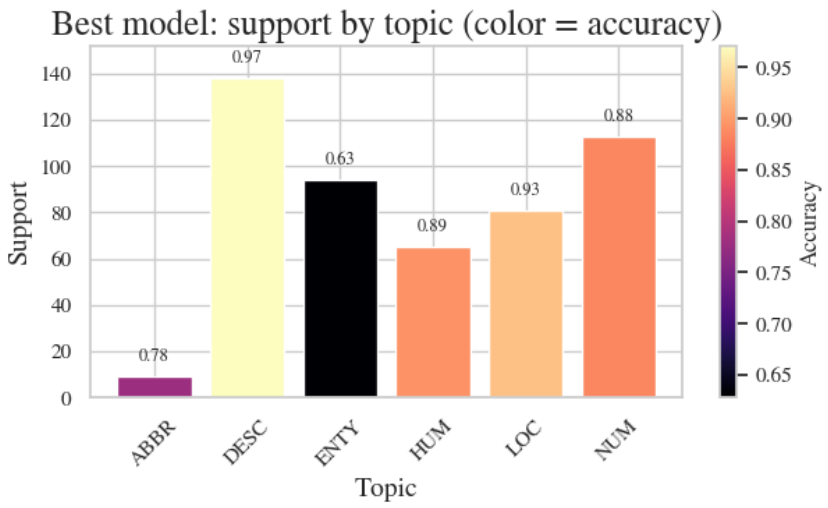

Figure 7: Topic-wise accuracy of the best RNN.

4 Enhancements

4.1 Bidirectional Recurrent Neural Networks (RNNs)

To improve the performance of the baseline RNN model, two bidirectional variants were used: a BiLSTM (Bidirectional Long Short-Term Memory) and a BiGRU (Bidirectional Gated Recurrent Unit) model. Both of these models extend the simple RNN model by incorporating recurrent computations in both forward and backward directions, allowing the model to capture contextual dependencies from both past and future tokens. This is particularly valuable in topic classification, where understanding a word often depends on both its preceding and succeeding contexts.

The BiLSTM model contains two LSTM layers, one that reads the input from start to end and another that reads it from end to start. Each LSTM cell decides what information to keep or forget, helping the model remember important details over long sequences. The resulting training loss and validation accuracy curves from applying this model can be referred to in Figure 8 below. The resulting test accuracy was 88.4%.

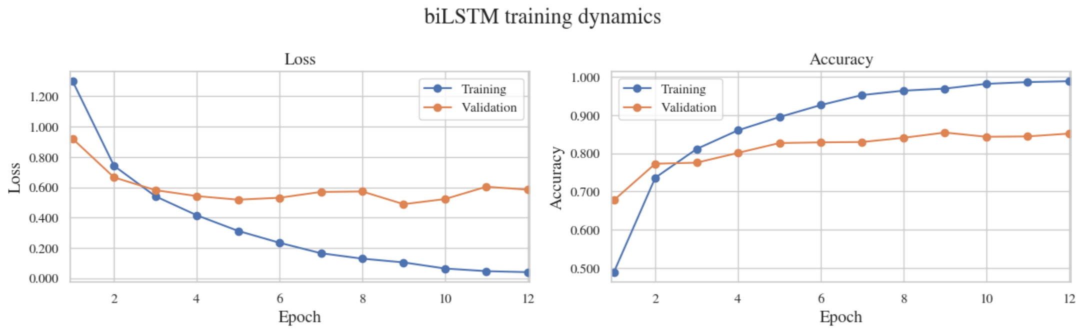

Figure 8: biLSTM training loss and validation accuracy curves.

The BiGRU model is a simplified variant of the BiLSTM that combines the input and forget gates into a single update gate and merges the hidden and cell states. This reduces the number of parameters while preserving the ability to model long-range dependencies. The resulting training loss and validation accuracy curves from applying this model can be referred to in Figure 9 below. The resulting test accuracy was 89.6%.

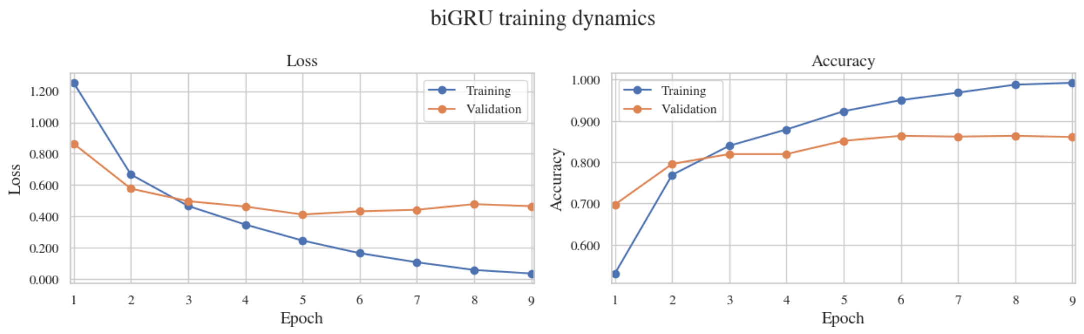

Figure 9: biGRU training loss and validation accuracy curves.

Both models show strong improvements compared to the baseline RNN, which had a test accuracy of 84.8%. The BiGRU slightly outperforms the BiLSTM, likely due to its simpler gating mechanism allowing for faster convergence and less overfitting.

4.2 Convolutional Neural Network (CNN)

The CNN model replaces the sequential recurrence of RNNs with 1D convolutional filters that act on word embeddings to detect local n-gram features. Each convolution kernel extracts phrase-level patterns (e.g., "climate change", "financial crisis"), and max-pooling selects the most important features across the sentence, producing a compact representation for classification.

The resulting training loss and validation accuracy curves from applying CNN can be referred to in Figure 10 below. The resulting test accuracy was 89.4%.

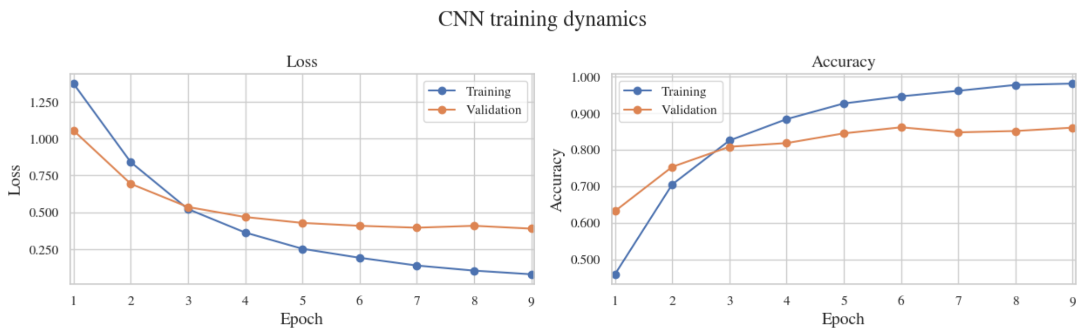

Figure 10: CNN training loss and validation accuracy curves.

4.3 Further Model Improvements

To further improve the performance of the topic classification model, a Recurrent Convolutional Neural Network (RCNN) was implemented. This architecture combines the sequential modeling of recurrent networks with the ability to extract local features from convolutional networks.

In this model, a biGRU first encodes each token into a contextual representation by capturing information from both preceding and succeeding words in the sentence. Each contextual vector is then concatenated with its corresponding word embedding, preserving both the original semantic meaning and the contextual information learned by the recurrent layer. Following this, a linear 1x1 convolution layer with a tanh activation acts as a feature transformation step, letting the model learn nonlinear interactions between contextual and semantic features. A masked global max-pooling operation is then applied across the sequence to extract the most significant features. The resulting pooled vector is passed through a fully connected layer for final classification. Multi-sample dropout (N=4) was applied to reduce variance.

The resulting training loss and validation accuracy curves from applying RCNN can be referred to in Figure 15 below. The resulting test accuracy was 92.2%.

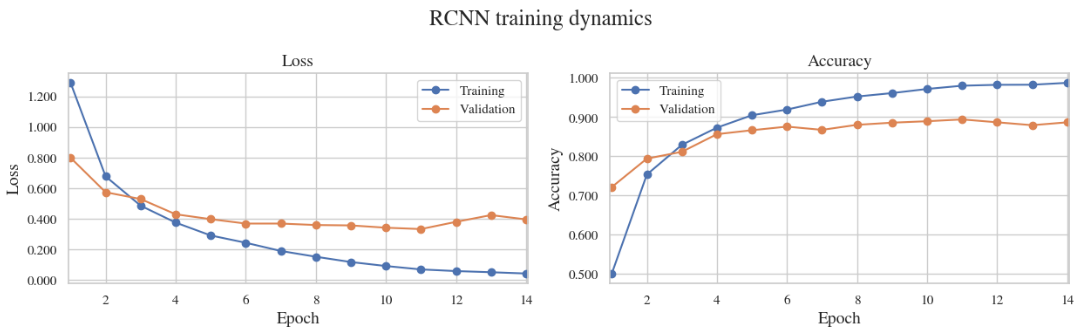

Figure 11: RCNN training loss and validation accuracy curves.

Compared with earlier models, RCNN effectively integrates RNN-based architectures, which capture long-range dependencies but often lose detailed local information, and CNN-based architectures, which excel at recognizing local patterns but lack a sense of sequence. By integrating both mechanisms, the RCNN is able to retain global context while emphasising the most informative word-level features.

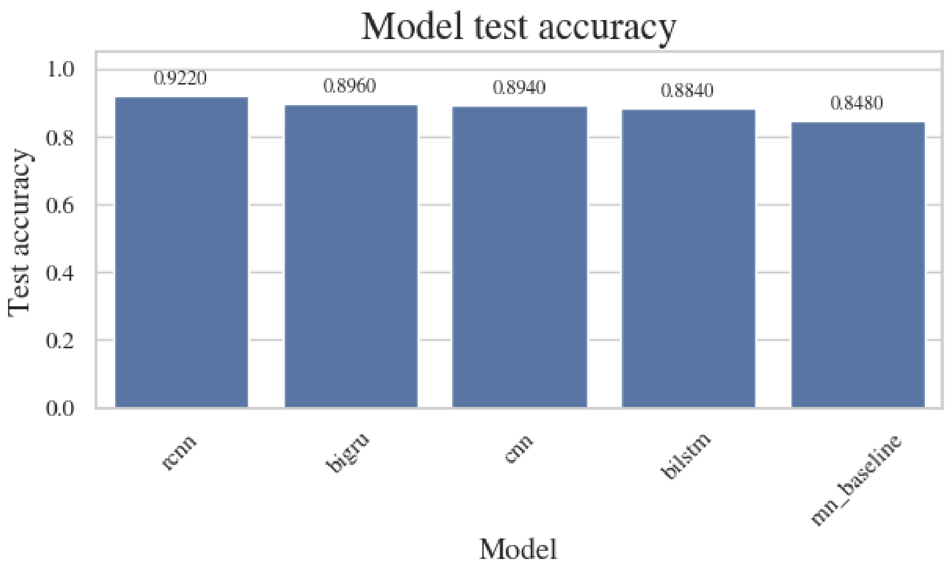

Figure 12: Test accuracy comparison between the models.

As seen above in Figure 12, RCNN resulted in performance improvement compared to RNN, BiGRU, BiLSTM, and CNN.

4.4 Targeted Improvement on Topics

From the previous sections, we have identified the two weak categories as ENTY and ABBR.

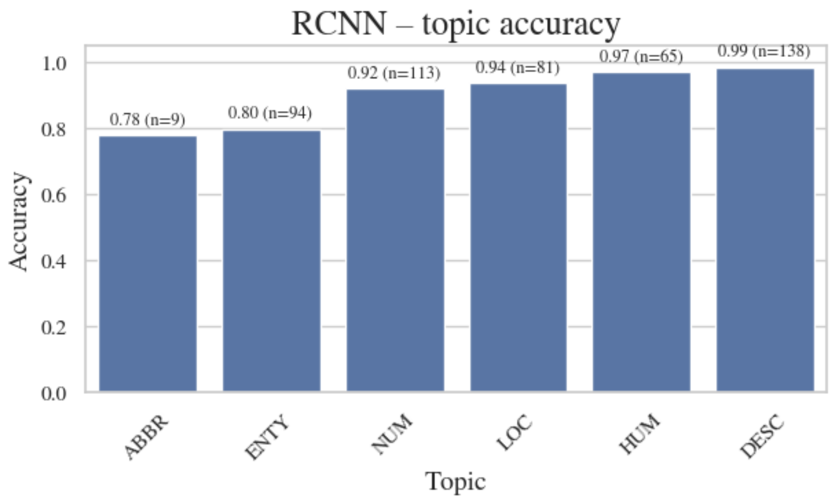

Figure 13: Original RCNN topic accuracies.

The underlying problem observed from the ENTY class was primarily the large number of subclasses that fragmented the training signal and caused confusion with semantically similar categories such as DESC and HUM. To address this, we implemented the RCNN model from the further improvement section (4.3) with the introduction of a contrastive auxiliary coarse topic loss in which enhances coarse-level learning by enforcing structured relationships in the model's embedding space.

For the ABBR class, the notable issue was the extremely shallow pool of samples (n=9), which made it difficult to generalize abbreviation patterns from the training corpus. In an attempt to resolve this, we applied a targeted date- and loss-based strategy using the RCNN backbone using In-batch oversampling. This dual approach would have increased the model's exposure to abbreviation patterns, allowing adjustment of decision boundaries around scarce examples.

The ENTY class had a significant improvement in the coarse ENTY accuracy following the implementation of the above improvement strategy, confirming that the setup effectively reinforced the ENTY class recognition without negatively affecting other categories.

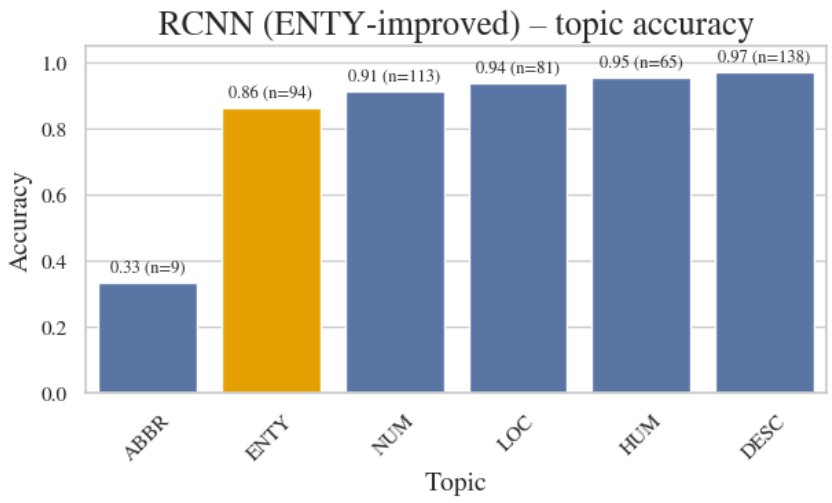

Figure 14: RCNN - ENTY improvement strategy applied.

However, for the ABBR class, there was no observable improvement in its accuracy, limited by the small sample size. The applied method may have redirected random misclassifications of ABBR samples into other categories but has had no tangible results in improving accuracy. Adding more samples or referring to external sources may be able to yield more noticeable results.

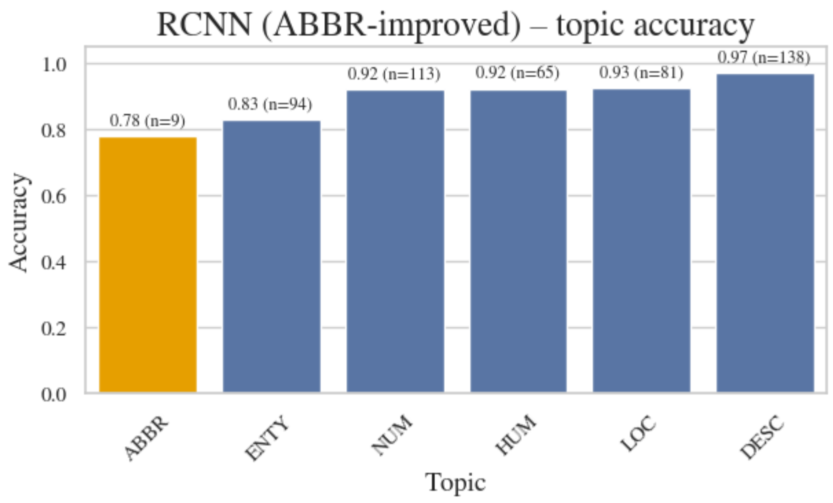

Figure 15: RCNN - ABBR improvement strategy applied.