WikiQA Open-Domain Question Answering using Bi-LSTM and Attention

CITS4012 Assignment Group 31

Abhishek Anand[1], Shaikh Enamul Haque[2] and Jia Min Ho[3]

University of Western Australia, Australia

[1]23598144@student.uwa.edu.au
[2]23440037@student.uwa.edu.au
[3]23337561@student.uwa.edu.au

1 Dataset

The Microsoft Research WikiQA Corpus is a dataset consisting of question and sentence pairs, designed for research in open-domain question answering [1]. The questions were derived from Bing query logs and are linked to Wikipedia pages that potentially provide the answers. The dataset comes with a training and testing dataset, both of which include attributes such as QuestionID, Question, DocumentID, DocumentTitle, SentenceID, Sentence, and Label (which indicates the answer sentence if label=1). We constructed three different types of data for training the model: Question, Document and Answer.

In our data preprocessing stage, we aimed to create an effective token labeling strategy that would help the model better identify the correct answers within the given documents. Initially, we attempted to label each token of the document into three token types: SOA (Start Token of the Answer), EOA (End Token of the Answer), and NONE. However, this approach led to an uneven distribution of tokens, causing the model to predict mostly NONE tokens. To overcome this issue, we modified our token labeling strategy to include four token types: BA (Before Answer), A (Answer), AA (After Answer), and PAD (Padding). The token labels are assigned based on whether the sentence contains the answer, comes before the answer, or appears after the answer. Since LSTMs take inputs of the same length and dimension, padding is applied to ensure all documents have the same length, making it easier for the model to process the input sequences [15]. By adopting this modified token labeling strategy, we create a more balanced representation of the data and enable our model to better focus on learning the relationships between words in a sequence and identify correct answers more effectively. This approach contributes to improving the overall performance of our model in the open-domain question-answering task.

Initially, we employed the maximum length for sequence padding in our model. However, this approach was not effective as the average document length is 219, while the maximum document length is 872. Padding all sequences to the maximum length can introduce a large number of padding tokens for shorter sequences, resulting in inefficiencies during training and causing the model to focus more on padding tokens than the actual content. To address this, we arbitrarily chose the maximum sequence length to be 219 (the average length of the sequences in the dataset) and forced all sequences to be of the same length by truncating the longer sequences and padding the shorter sequences with PAD token. This results in a more balanced data representation, faster training, and improved generalization [16]. For training data, we followed a specific approach to ensure that only documents with answers were included. This selection criterion helps the model focus on learning from relevant and informative instances during the training process. However, for the testing dataset, we added all the documents to the list without any filtering based on the presence of answers. This allows us to evaluate the model's performance comprehensively on a wider range of documents, including those without explicit answers.

2 Sequence QA model

2.1 **Word Embedding**

FastText, developed by Facebook AI Research [2], is a popular choice for pre-trained word embeddings and offers certain advantages over other methods such as Word2Vec (CBOW and Skip-gram) and GloVe. FastText is particularly well-suited for our specific use case for several reasons. First, FastText takes into account the internal structure of words by representing them as a bag of character n-grams [9]. This allows the model to capture subword information, which can be particularly useful for handling morphologically rich languages, rare words, and out-of-vocabulary (OOV) words. Second, unlike Word2Vec and GloVe, FastText can generate embeddings for OOV words by summing the character n-grams' embeddings. This means that even if a word is not present in the pre-trained vocabulary, FastText can still generate a meaningful representation for it, which can be beneficial for many NLP tasks. Lastly, FastText is designed to be computationally efficient, both in terms of memory usage and training time. This allows us to train embeddings on large-scale datasets relatively quickly, even on modest hardware.

2.2 **Feature Extraction**

We have chosen PoS Tags, TF-IDF, and Named Entity Tags (NER) as our feature embeddings for this project, as these three features can be particularly useful for open-domain question-answering tasks. Part-of-speech (PoS) tagging helps capture the grammatical structure of sentences, allowing the model to understand the syntactic roles of words and discern the relationships between them [6]. Incorporating PoS tags enables the model to identify key elements, disambiguate words with multiple meanings, and comprehend the context of questions and potential answers more effectively.

The TF-IDF (term frequency, inverse document frequency) representation is an effective method for identifying the importance of words within a document relative to a corpus. By using TF-IDF features, our model can assign higher weights to words that are more important or relevant to a specific document and lower weights to common words that appear across multiple documents [5]. This helps the model focus on the most relevant information in the text, thereby improving its ability to identify correct answers.

Named Entity Recognition (NER) is a technique used to identify and classify real-world objects such as people, organizations, locations, and dates within the text [4]. Including Named Entity Tags as features helps our model pay more attention to these specific entities, which often hold crucial information related to the questions being asked. By focusing on these entities, the model can better understand the relationships between them and extract relevant answers from the text.

By combining these three types of features, we provide our model with a rich representation of the text, enabling it to capture both the semantic and syntactic aspects of the data. This combination helps the model focus on essential information, identify relationships between words and entities, and better understand the context of the questions and potential answers.

2.3 **Sequence QA Model (Recurrent Model with Attention)**

2.3.1 **Sequence QA Model: Detailed Architecture of Model**

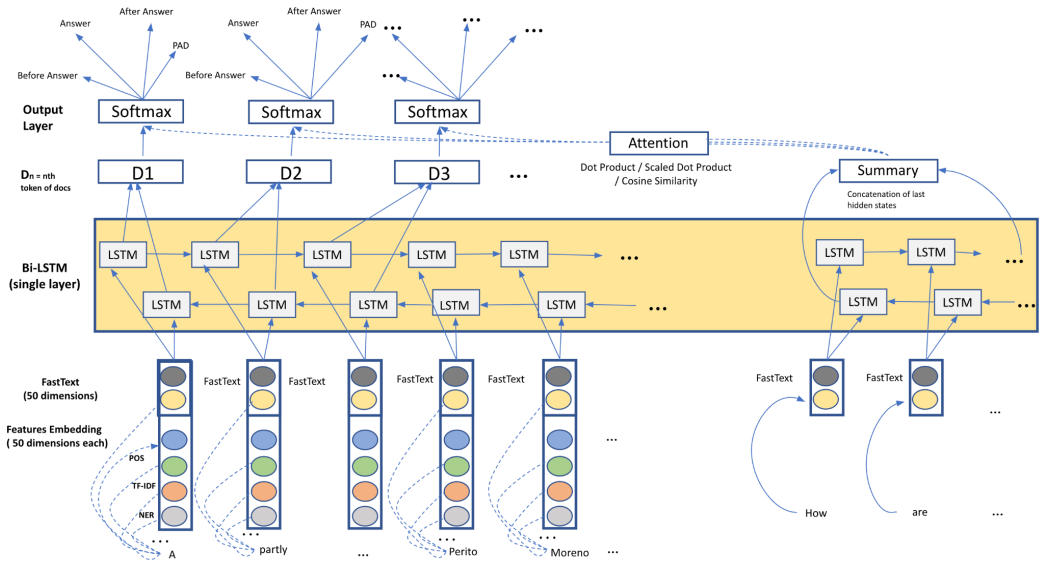

Figure 1: Detailed Architecture of Bi-LSTM with Attention Model for WikiQA Open-Domain Question Answering.

For sequence modeling in the WikiQA Open-Domain Question Answering task, we selected the Bi-LSTM architecture due to its ability to capture long-range dependencies, prevent the vanishing gradient problem, and effectively comprehend past and future context in input sequences [3, 7]. Bi-LSTM combines the strengths of LSTM, which uses gating mechanisms to retain or forget information in each state, addressing the challenge of preserving long-term information and overcoming the vanishing gradient problem encountered by traditional RNNs [14]. The bidirectional nature of Bi-LSTM, consisting of two separate LSTM networks processing sequences in both forward and backward directions, further enhances the model's understanding of text by considering a comprehensive context. These features are particularly valuable for open-domain question-answering tasks, where understanding context and relationships within the text is vital for accurate predictions.

For this task, we implemented two separate Bi-LSTM networks: one for encoding the question (BiLSTMForQuestion) and another for encoding the document (BiLSTMForDocument). In BiLSTMForQuestion, we utilize the last hidden state of the entire sequence as the question summary. This summary captures the contextual information of the question, which is then used in the attention mechanism for the document encoding. The BiLSTMForDocument includes an attention mechanism, allowing the model to weigh the importance of each token in the document based on the question's encoded summary. The attention mechanism, positioned after the Bi-LSTM layer, helps the model focus on the most relevant parts of the input sequences, effectively capturing relationships between words and improving performance in identifying correct answers [7, 13]. The attention mechanism operates by computing a weighted sum of the Bi-LSTM hidden states, with the weights representing the importance of each word in the document concerning the question. This approach enables the model to concentrate on the most significant parts of the input sequences when predicting the answer, rather than treating all words in the document equally. The model's output is a softmax function predicting a single class as the start token. This softmax layer assigns probabilities to each token in the document, indicating the likelihood of it being the beginning of the answer span. By selecting the token with the highest probability, the model pinpoints the most likely starting point for the correct answer and we will select that sentence as our answer.

We opted for CrossEntropyLoss over Negative Log-Likelihood Loss in this task, as CrossEntropyLoss expects raw prediction values while NLLLoss expects log probabilities [8]. The implementation of CrossEntrypyLoss implicitly applies a softmax activation followed by a log transformation but NLLLoss does not. Therefore, CrossEntropyLoss offers a more streamlined and efficient approach for our purposes. Finally, after experimenting with both the SGD (Stochastic Gradient Descent) and Adam optimizers, we selected the latter for training the model. This decision was based on the observed faster convergence and improved performance of the Adam optimizer, which is attributed to its adaptive learning rate and reduced noise in gradient updates, as evidenced in numerous studies [10, 11, 12].

2.3.2 **Sequence QA Model: Optimal Number of Layers**

Model Type	Model Description	Precision Score	Recall Score	F1 Score
Input Embedding Ablation Study	FastText only. No TFIDF, POS and NER tags	0.175847	0.344398	0.232819
Input Embedding Ablation Study	FastText with TFIDF and POS tags. No NER tag	0.167024	0.323651	0.220339
Input Embedding Ablation Study	FastText with TFIDF, POS and NER tags	0.148796	0.282158	0.194842
Attention Ablation Study	FastText only. Dot Product Attention	0.17234	0.3361	0.227848
Attention Ablation Study	FastText only. Scaled Dot Product Attention	0.165236	0.319502	0.217822
Attention Ablation Study	FastText only. Cosine Similarity Attention	0.167024	0.323651	0.220339
Hyper Parameter Testing	FastText only. 500 Epochs, 0.01 Learning Rate	0.197938	0.39834	0.264463
Hyper Parameter Testing	FastText only. 1000 Epochs, 0.001 Learning Rate	0.182773	0.360996	0.242678
Hyper Parameter Testing	FastText only. 1000 Epochs, 0.1 Learning Rate	0.256214	0.556017	0.350785
Hyper Parameter Testing	FastText only. 5000 Epochs, 0.1 Learning Rate	0.26465	0.580913	0.363636
Hyper Parameter Testing	FastText only. 10000 Epochs, 0.1 Learning Rate	0.266038	0.585062	0.365759
Number of Layers - 5	FastText only. 5000 Epochs, 0.1 Learning Rate	0.266038	0.585062	0.365759

Table 1: Performance Evaluation of Different Model Variants, Hyperparameter Configurations and Number of Layers.

Table 1 displays the results of the model testing, including various input embedding ablation studies, attention ablation studies, hyperparameter testing and number of layers testing. The precision, recall, and F1 scores are presented for each model configuration. After conducting an extensive evaluation of different numbers of layers for our sequence model, we have determined that using only one layer is the optimal choice. This decision was based on a thorough analysis of the model's performance, computational complexity, and generalization capabilities.

Initially, we performed an ablation study on input embeddings, attention mechanisms and hyperparameter testings to identify the most effective configurations using only one layer. Among the evaluated models, one particular model stood out as the best performer with an F1 score of 0.365759 (Model 11). We then selected this model as the baseline to compare the impact of different numbers of layers on performance. When we tested the model with 5 layers (Model 12), we observed that its performance remained virtually unchanged compared to the one-layer model. The precision, recall, and F1 scores were identical, indicating that additional layers did not provide any noticeable benefit in terms of predictive accuracy. However, it is crucial to consider the computational complexity of the model. Increasing the number of layers leads to a more complex architecture, which can significantly impact training time and memory requirements [20]. In our experiments, we observed that the 5-layer model required considerably more computational resources and time compared to the one-layer model. Furthermore, generalization was a key consideration in our decision-making process. Adding more layers to the model has the potential to increase its capacity to fit the training data more closely. However, it also raises the risk of overfitting, where the model becomes too specialized in capturing noise and idiosyncrasies within the training set, leading to poor performance on unseen data. Considering all these factors holistically, we determined that using one layer strikes the best balance between performance, complexity, and generalization. The one-layer model achieves a satisfactory F1 score, maintains computational efficiency, and avoids the risk of overfitting. Additionally, it offers faster computation and proves to be cost-efficient, making it the preferred choice for our sequence model.

3 Model Testing

In our WikiQA open domain question answering task, precision, recall, and F1-score are critical evaluation metrics for assessing the performance of input embeddings, attention types, and hyperparameters. Precision ensures the delivery of accurate and relevant answers by measuring the proportion of correct positive predictions. Recall, on the other hand, captures the model's ability to retrieve relevant answers by evaluating the proportion of correct positive predictions out of all positive examples. The F1-score combines precision and recall into a single metric, providing a balanced assessment of the model's overall performance. These metrics are crucial in delivering accurate and comprehensive answers to user queries. Achieving a balance between precision and recall is vital for the overall effectiveness of our model in providing precise and relevant answers in an open domain context.

3.1 Input Embedding Ablation Study

The performance of different input embedding variants, including FastText only, FastText with POS tag embedding, and FastText with all three features embeddings, was assessed using precision, recall, and F1-score metrics. The results of this evaluation are presented in the table below:

Input Embedding Variant	**Precision**	**Recall**	**F1-score**
FastText only	0.175847	0.344398	0.232819
FastText + TF-IDF + POS Tag	0.167024	0.323651	0.220339
FastText + TF-IDF + POS + NER Tag	0.148796	0.282158	0.194842

Upon analyzing the results, it was evident that the FastText only embeddings exhibited the highest performance among the evaluated input embedding variants. It achieved better precision, recall, and F1-score compared to the other variants. This outcome suggests that relying solely on FastText embeddings is sufficient for capturing and comprehending relevant information, leading to improved predictive accuracy and completeness of answers. The inclusion of additional features such as POS tag, TF-IDF, and NER embeddings did not contribute significantly to the model's performance. In this particular open domain question answering task, the FastText only approach outperformed the variants incorporating additional feature embeddings. These findings emphasize the importance of thoroughly evaluating and considering different input embedding strategies in order to determine the optimal configuration for a given task.

3.2 **Attention Ablation Study**

To evaluate the impact of different attention mechanisms on our model, we conducted an ablation study. We tested three variants of attention calculations: Dot Product, Scaled Dot Product, and Cosine Similarity attention. We chose these 3 variants because they provide a balance between simplicity, efficiency and effectiveness for open-domain question-answering tasks. They provide a strong focus on similarity between document tokens and question summary. These methods are also computationally less expensive than others like content-based, additive/concat, location-based, or general attention mechanisms. Dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code, making it better than additive attention [17].
Scaled Dot Product is a modified version, dividing the dot product result by the square root of the key's dimension to stabilize gradients during training and prevent softmax saturation in high-dimensional spaces [17]. Cosine Similarity calculates the cosine of the angle between query and key vectors, focusing on their orientation rather than magnitude, making it suitable for determining the relation of a word to the question in question-answering tasks [18].

The performance of each variant was evaluated in terms of precision, recall, and F1-score. The results are visualized in the table below:

Attention Variant	**Precision**	**Recall**	**F1-score**
Dot Product	0.17234	0.3361	0.227848
Scaled Dot Product	0.165236	0.319502	0.217822
Cosine Similarity	0.167024	0.323651	0.220339

We observed that the Dot Product attention variant performed the best among the three, with the highest precision, recall, and F1-score values. This could be attributed to several factors. Firstly, the Dot Product operation is a simpler and more straightforward method compared to the scaled version or cosine similarity. The simplicity and efficiency of the Dot Product make it easier for the model to learn meaningful attention weights and capture the relevant relationships between the question and the document tokens. Furthermore, the Dot Product attention variant may have benefited from the specific characteristics of the dataset or the nature of the task. It is possible that the dot product operation aligned well with the underlying patterns and dependencies present in the data, leading to more accurate predictions and better performance in the WikiQA Open Domain Question Answering Task.

3.3 **Hyper Parameter Testing**

To optimize the performance of our model, we conducted an evaluation of various hyperparameter configurations, specifically focusing on the number of epochs and learning rates. Hyperparameters play a crucial role in training machine learning models as they control the optimization process and influence the model's ability to learn and generalize from the data. By systematically testing different combinations, we aimed to identify the configuration that yielded the best results. In our experimentation, we tested five different hyperparameter variants, each with a unique combination of epochs and learning rates. The performance of each variant was assessed using precision, recall, and F1-score metrics.

Hyperparameter		**Precision**	**Recall**	**F1-score**
Epochs	**Learning Rate**			
500	**0.01**	0.197938	0.39834	0.264463
1000	**0.001**	0.182773	0.360996	0.242678
1000	**0.1**	0.256214	0.556017	0.350785
5000	**0.1**	0.26465	0.580913	0.363636
10000	**0.1**	0.266038	0.585062	0.365759

Among the tested variants, the fifth configuration, utilizing 10000 epochs and a learning rate of 0.1, emerged as the best performing one. The choice of a higher learning rate of 0.1 facilitated faster convergence during training, allowing the model to make significant weight updates in shorter intervals. This acceleration in learning enabled the model to effectively capture the underlying patterns and relationships in the data [19]. Furthermore, the high number of epochs (10000) allowed the model to extensively explore the solution space, learn nuanced features, and continuously refine its understanding of the data throughout the extended training process. It's important to note that while this approach requires a longer training time, it ultimately contributed to the achievement of superior performance on the chosen evaluation metrics. This optimal combination of learning rate and number of epochs resulted in a well-tuned model that achieved the highest overall performance.

References

1. Yang, Y., Yih, W., Meek, C.: WikiQA: A Challenge Dataset for Open-Domain Question Answering. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2013-2018. Association for Computational Linguistics, Lisbon, Portugal (2015).

2. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T.: Enriching Word Vectors with Subword Information. Facebook AI Research (2016).

3. Jang, B., Kim, M., Harerimana, G., Kang, S.U., & Kim, J. W.: Bi-LSTM Model to Increase Accuracy in Text Classification: Combining Word2vec CNN and Attention Mechanism. Applied Sciences 10(17), 5841 (2020).

4. Belainine, B., Fonseca, A., Sadat, F.: Named Entity Recognition and Hashtag Decomposition to Improve the Classification of Tweets. In Proceedings of the 2nd Workshop on Noisy User-generated Text, pp. 102–111. The COLING 2016 Organizing Committee, Osaka, Japan (2016).

5. Jing, L., Huang, H., Shi, H.: Improved feature selection approach TFIDF in text mining. In Proceedings. International Conference on Machine Learning and Cybernetics, vol. 2, pp. 944-946. Beijing, China (2002).

6. Chiche, A., Yitagesu, B. Part of speech tagging: a systematic review of deep learning and machine learning approaches. Journal of Big Data 9(10), (2022).

7. Chen, D., Bolton, J., Manning, C.: A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 2358-2367. Berlin, Germany (2016).

8. Zhang, Z., Sabuncu, M.: Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. 32nd Conference on Neural Information Processing Systems (NeurIPS) (2018).

9. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135-146 (2017).

10. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. International Conference on Learning Representations, 1–13 (2015).

11. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization, in Proceedings of International Conference on Learning Representations, Vancouver, Canada, (2018).

12. Wilson, A., Roelofs, R., Stern, M., Srebro, N., Recht, B.: The marginal value of adaptive gradient methods in machine learning. In Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 4151-4161, Long Beach, CA, USA (2017).

13. Luong, M., Pham, H., Manning, C.: Effective Approaches to Attention-based Neural Machine Translation. Computer Science Department, Stanford University, Stanford, CA (2015).

14. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation, 9(8), 1735-1780 (1997).

15. Dwarampudi, M., Reddy, N.: Effects of padding on LSTMs and CNNs. arXiv, 1903.07288 (2019).

16. Lu, H., Ehwerhemuepha, L., Rakovski, C.: A comparative study on deep learning models for text classification of unstructured medical notes with various levels of class imbalance. BMC Med Res Methodol, 22(1), 181 (2022).

17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems pp. 5998-6008. (2017).

18. Luo, C., Zhan, J., Wang, L., Yang, Q.: Cosine Normalization: Using Cosine Similarity Instead of Dot Product in Neural Networks. In Artificial Neural Networks and Machine Learning, pp. 382-391 (2018).

19. Adedigba, A., Adeshina, S., Aina, O. E., Aibinu, A.: Optimal hyperparameter selection of deep learning models for COVID-19 chest X-ray classification. Intelligent Based Medicine, 5, 100034 (2021).

20. Li, H., Wang, Z., Yue, X., Wang, W., Tomiyama, H., & Meng, L.: An architecture-level analysis on deep learning models for low-impact computations. Artificial Intelligence Review, 56, 1971-2010 (2023).