Automated Evaluation of Handwritten Answer Script Using Deep Learning Approach

Md. Afzalur Rahaman[1], *Hasan Mahmud*[2]
Department of CSE, Hamdard University Bangladesh[1,2]

August 16, 2022

1 Abstract

Automatic Essay Grading (AEG) is one of the exciting research topics in the field of adopting technology in education. In the education system assessment of student's answer script is a critical job of teachers; yet doing so consumes a significant amount of their time and prevents them from working on other tasks. In addition, evaluating a large number of exam scripts is error-prone, inefficient, and tedious. Natural Language Processing (NLP), has created such an opportunity to make the computer learn about written text data and make important decisions based on the learned model. Similarly, it is possible to make a computer be able to assess an answering script based on the model used to train our computer to learn about answers to predefined short questions. In this paper, we propose a deep learning architecture with a combination of Convolutional Neural Network (CNN) and Bidirectional Long Short Term Memory (BiLSTM) which has the ability to perform both handwritten answers recognition and grading them as accurately as a human expert grader.

Keywords— Automatic Script Grading, Bidirectional LSTM Network, Convolutional Neural Network, Deep Learning, Natural Language Processing, Word Embedding

2 Introduction

Since the last four decades, computer is using for writing essay and to assess automatic submissions of students work homogeneously. The best pedagogy for improving students writing skills is to check each submission for each student and to replay them individually by a teacher in the classroom. Unfortunately, this significantly increases the workload for teachers. In fact, responding to student papers and rigorously checking them is a burden for many teachers and this pressure linearly increases with the amount of students increases. Therefore, developing an automated system can help to reduce the cost of checking in a significant way and facilitates students to get early feedback. Handwritten text recognition is the ability of a computer to interpret a text

from sources like scripts, images, or others. The image is scanned optically with handling formats, segmenting lines and words into characters to trace the most plausible characters. The most challenging problem related to handwriting recognition is recognizing different styles and sizes with a good accuracy level. The aim of this study is to explore the task of classifying handwritten text and to convert them into digital format and to grade them automatically. The development of AEG was mostly started with Latent Semantic Analysis (LSA), N-Gram, TF-IDF, Bayesian classifier, and K-nearest neighbor approaches, although the satisfactory performance level wasn't achieved well. After the revaluation of Deep Learning (DL) and Natural Language Processing (NLP), much research has been done on the automatic evaluation of computer-based submitted essays and higher accuracy is gathered. In contrast, a few amount of researches have been done on Handwritten Essay Grading (HEG) and no model is developed to be a replica of an expert human grader. Till now much research is being done in this sector and researchers investigating machine scores to move forward in their drive with the aim of improving the accuracy and effectiveness of the system. The proposed system is intended to be an aid for grading a large corpus of student's answers, reducing the teacher's load, thereby encouraging them to give students more writing essay tasks.

The rest of the paper is organized as follows: section III presents the relevant works on AEG systems. Section IV describes the overall framework of the proposed system. Section V presents the performance analysis and discussion. Finally, conclusion and future work are drawn in section VI.

3 Relevant works

The development of an automatic essay system is not an over night invention. The first driving force behind handwritten text classification was for digit recognition for postal mail. Allum et al. improved a sophisticated scanner which recognizes how the text was written as well as encoding the information onto a bar code.
The first prominent piece of OCR software was developed by Ray Kurzweil in 1974 as a software to recognize font. The software was sensitive to variations in sizing and the distinctions between each individuals way of writing. The next major upgrading on OCR accuracy was made by the use of a Hidden Markov Model with the task of OCR. This approach uses letters as a state.
The robust architecture of neural network replaced the traditional feature-extracting methods with the excellent one. Neural networks is a mathematical models that mimic the structure and function of biological neural networks. In most cases with the fast adjustment of parameters, a neural network can learn very fast and performs excellent on test set. In recent decades, with the deepening of machine learning and artificial intelligence research, neural network models are introduced in the AES system.
Among them Dong and Zhang [1] used a dense convolutional neural network model for essay scoring. The input is given with word level convoluational layer followed by a word embedding layer. From the two layer CNN architecture one convolutional layer is used to extract sentences representations, and the other is stacked on sentence vectors to learn essays representations. The model is trained and evaluated with the Automated Student Assessment Prize (ASAP) dataset and achieved indomain average kappa score close to human graders.
Alikaniotis et al. [2] (2016) employed a neural model to learn features for essay scoring automatically, which leverages a score-specific word embedding (SSWE) for word rep-

resentations and a two-layer bidirectional long-short term memory network (LSTM) to learn essay representations. Alikaniotis et al. showed that a model with the combination of SSWE and LSTM outperforms compared to traditional SVM based model. Where standalone LSTM does not give significant accuracy compared to SVM based model.
For classifying words and character segmentation B. Balci et al. [3] used a CNN with various architectures to train a model that can accurately classify words. To construct bounding boxes for each character they also used LSTM network with the convolution. T.Wang et al. [4] applied CNNs to the problem and identified text within the image by using a sliding window. The sliding window moves across the image to find a potential instance of a character being present. A CNN with two convolutional layers, two average pooling layers, and a fully connected layer is used to classify each character.
One of the most prominent paper for the task of handwritten text recognition is developed by Bluche et al [5]. The approach used an LSTM layer for each scanning direction and encode the raw image data to a feature map. The model then used attention to emphasize certain feature maps over others. The attention map is given into the decoder which then predicts the character.
A. Shehab [6] developed an automated essay grading system based on writing feature analysis that evaluates and feedbacks for errors in grammar, identifies the essays discourse structure, and recognize undesirable stylistic features using NLP techniques. The grading engine is trained on a set of pre-graded essays to grade a new essay. The model obtained 70% to 90% closeness between human and machine assigned grades.
The Auto-mark software system is developed for a robust computerized marking of open ended questions [7]. It uses NLP techniques to grade open-ended responses, where a submitted text is pre-processed to standardize the input in terms of punctuation and spelling. A sentence analyzer then identifies the main syntactic constituents of the text and how they are related. Finally, the feedback module processes the result of the pattern matched.
C. Cai [8] developed a model on feature analysis and RNN. The feature analysis is done by checking spelling errors, number of unique vocabulary, punctuation, and unique nouns. The correlation between length of an essay and it's corresponding score is also analyzed by support vector regression (SVR) and Bayesian linear ridge regression (BLRR) approaches.

In 2012, a competition on automated essay scoring called 'Automated Student Assessment Prize' was organized by Kaggle. Later on, this dataset became much popular for NLP-based model research and development. Many researchers have devoted a substantial amount of effort to design an efficient scoring approach. The winning team got 81.407% similarity between the human scores and the automated system. Later, a team at Carnegie Mellon University built a model with dense and sparse features, trained on the same dataset, and achieved a 0.833 kappa score. Later on, H. Nguyen and L. Dery [9] developed a bidirectional LSTM model on the same dataset and achieved a 0.944 kappa score. In 2016, D. Alikaniotis et al. [2] developed another bidirectional LSTM model and achieved 0.96 kappa score, which indicates excellent performance is already achieved on computer-based essay submissions. In contrast, a few amounts of development was done on automatic HEG systems like, S. Srihari et al. [10] (2008) developed an essay grading system using latent semantic analyzer and ANN. A. A. A. Ali and S. Mallaiah [11] developed an intelligent handwritten recognition system with hybrid CNN and SVM. A. Sharma and D. B. Jayagopi [12] developed a system with Multi-Dimensional Long Short Term Memory (LSTM) and convolution

layers. Most existing systems are mainly developed for handwritten text recognition only and no complete HEG system is developed yet with a good performance level to be a replica of a human grader, which indicates much more research work is left to be done. To address the issues, we presented an automatic evaluation system, the architecture of which is briefly detailed in the next section.

4 Model Development

Handwritten text recognition can be defined as identifying spatial form into its symbolic representation. For the work in this paper, we first formulate the problem and then described the architecture of the proposed system with their background components like Convolutional Neural Networks, NLP, recurrent Long Short term Memory, and Bidirectional Long Short term Memory networks.

Convolutional Neural Network (CNN): The CNN is a class of multilayer neural networks specifically designed to process two-dimensional data. CNN is regarded as a successful deep-learning approach with a multi-layer hierarchical network. The generalization ability of CNN is significantly better than other methods. Mostly, CNN is used in image processing like image identification and classification. CNN have the ability to take image as input data and analyze that data to extract different features and also can make decision based on the extracted features. The CNN approach takes an input image and gradually extracts different features using multiple building blocks like convolution layers, pooling layers, and fully connected layers. The convolutional layers transform the input image into a stack of filtered images to extract features using filters or kernels. The activation layer is for aggregating the output of different layers for the next convolution process. Polling layers are then used to downsample the dimensions of features from convolutions to improve computational performance. The outputs of the convolutional and pooling layers finally entered the fully connected layer. The CNN is extensively applied in pattern classification, object detection, and object recognition.

Long Short Term Memory (LSTM): In many practical applications, data is interdependent. For example, when we understand the meaning of a sentence, it is not enough to understand each word independently; rather, we need to deal with the entire sequence of the words. When we think about a problem, we consider the thoughts of past experience and then combine it with the current situation. For the sequential problems, if we use a feedforward neural network, there are significant limitations since it can't remember the past information and is prone to vanishing gradient problems while backtracking. To overcome this limitation, RNN is extensively used for NLP since it handles long sequence dependencies well.
RNN has a very powerful application, which can successfully perform sequence tasks in many fields, such as speech recognition, robot translation, human-machine dialogue, speech synthesis, language modeling and translation, speech synthesis, and many other deep learning tasks. LSTM is an advanced development of RNN architecture where the output at time t is conditioned on the inputs both at time t and at time *t-1*. It is

a single layer architecture where we give total input at different time axes.

$$\begin{aligned}
&Forget\ gate\ f_t : \sigma(W_f \cdot [x_t, h_{t-1}] + b_f) \\
&Input\ gate\ i_t : \sigma(W_i \cdot [x_t, h_{t-1}] + b_i) \\
&Candidate\ cell\ \tilde{C}_t : tanh(W_c \cdot [x_t, h_{t-1}] + b_c) \\
&Update\ cell\ C_t : f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\
&Output\ gate\ o_t : \sigma(W_o \cdot [x_t, h_{t-1}] + b_o) \\
&hidden\ cell\ h_t : o_t \odot tanh(C_t)
\end{aligned} \tag{1}$$

where, x_t is the input layer at the current time step t , h_t is the value of the hidden layer of LSTM, while h_{t-1} denotes output values by each memory cell in the hidden layer at the previous time. The σ represents a sigmoid function, $\odot$ and tanh is element-wise multiplication and hyperbolic tangent function, respectively. The word vector W_i enter the input layer one at a time.

Bidirectional LSTM: The bidirectional LSTM proposed by Schuster and Paliwal is an extension to the traditional LSTM. In this case, current information has past information as dependencies and is also linked to future information. Unidirectional LSTM processed only the past information, which is prone to lose the actual information about a sentence. For example, our interpretation of a word at some point t_i might be different once we know the word at t_{i+n}. An effective way to get around this issue by training LSTM in a bidirectional manner. It has been proved that bidirectional networks are considerably better than unidirectional ones in many fields. The aim doing forward and a backward pass of the sequence (i.e., feeding the words from left to right and from right to left) to capture past and future information, respectively. A bidirectional LSTM has two distinct hidden layers, called the forward hidden layer $\overrightarrow{h}$ that considers the input in ascending order, and the backward hidden layer $\overleftarrow{h}$ considers the input in descending order. The two directions of the network act completely independently until the final layer, at which point their outputs are concatenated, can be shown as follows:

$$\begin{aligned}
\overrightarrow{h_t} &= g(W_{\overrightarrow{h}} x_t + W_{\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h_t}}) \\
\overleftarrow{h_t} &= g(W_{\overleftarrow{h}} x_t + W_{\overleftarrow{h}} \overleftarrow{h}_{t-1} + b_{\overleftarrow{h_t}}) \\
y_t &= g(W_{\overrightarrow{h}} \overrightarrow{h}_t + W_{\overleftarrow{h}} \overleftarrow{h}_t + b_y)
\end{aligned} \tag{2}$$

4.1 Handwritten Text Recognition

We have partitioned the research work into two phases: (i) Handwritten text (answers) recognition and (ii) Answers evaluation.
To recognize handwritten image text we developed an optical character reader (OCR) model using the IAM dataset. The whole process of segmenting the IAM dataset is shown in figure 1 and the functionalities of different phases is described are follows:

Pre-processing IAM dataset: The choice of suitable methodologies for our model is followed by the preprocessing actions on the IAM dataset. The actions include dimensional reduction, normalization, and inconsistency removal. Those actions help to create suitable image data to segment them easily. We used an OpenCV library in python to preprocess the image data. Preprocessing of image data is done based on the following three processes.

Noise Elimination: Sometimes an input image may have different types of spots

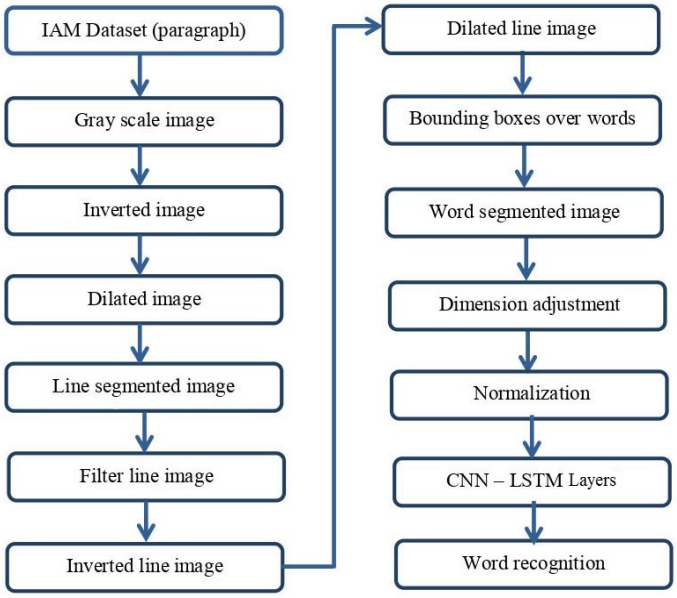
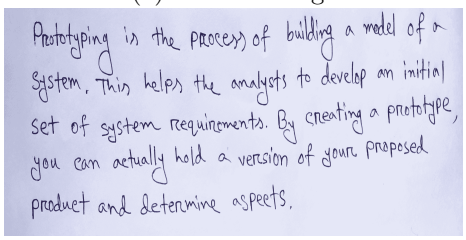


Figure 1: Handwritten text recognition model

which is not a part of the handwritten data. These type of spots are considered noise and have no influence on training the model properly and can be problematic while testing our model. So, before training our model with the dataset, we should remove those noises from the dataset.

Segmentation: There are two types of segmentation is involved in our preprocessing of data. One is known as Line Segmentation and another one is known as Word Segmentation. In our project, we are going to train our model with a number of single words. That means, single images of words will be input for our model. Hence, we need to perform word segmentation on the IAM dataset to create individual images of words. Since the IAM dataset is a dataset of many paragraphs, we have to do line segmentation first on the original dataset which are the collection of different para-

(a) Answer image



(b) Gray scaled answer

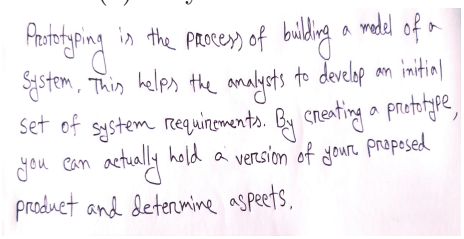


Figure 2: OCR steps visualization (1)

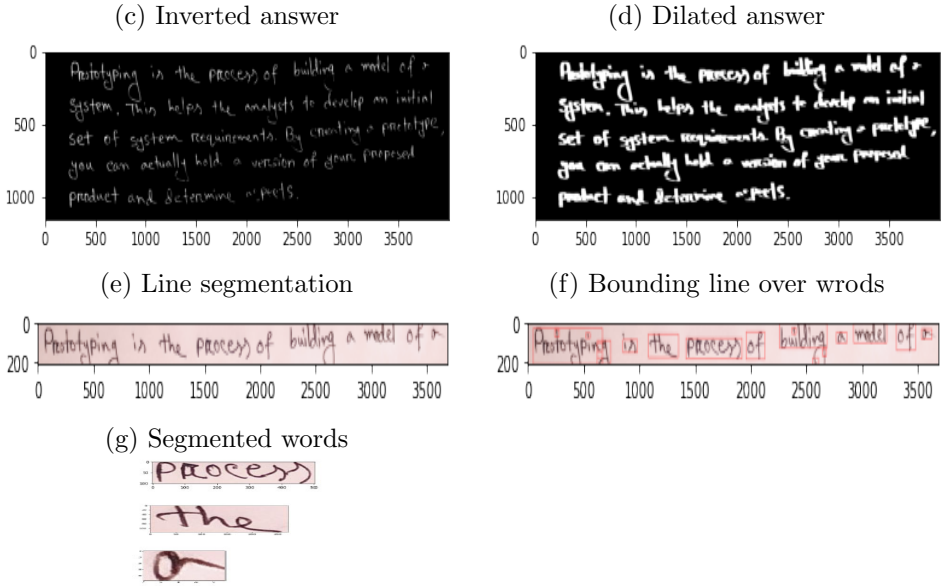


Figure 3: OCR steps visualizaiton (2)

graph. To create the image of single words for our model, we have to perform word segmentation process on the line segmented image data. At first, we have to create a gray scale image of original data. Using this gray scale image, we created an inverted image and later we converted it as a dilated image. Dilated image can be used to detect only the written portion on an image. From this dilated image, we can easily create bounding box over the each lines of our hand written text which will eventually generate line segmented image of the IAM dataset which was in a paragraph format. From those line segmented images we have to create word segmented image. The same procedure of making line segmented image have to be applied on the line segmented image. That means, at first we have to separate each words using the bounding box over each words of line segmented image. To do so, each line segmented image must be dilated, since we can identify the hand written text area on an image using dilated image. Let's generate a filtered line image out of the line segmented image data. These filtered line images are used to create inverted line image which are converted into dilated line image to create word segmented image for our training model. The whole process of segmenting the IAM dataset is shown in Figure 2 and 3.

Normalization: Our model has the ability to work with various sizes of fonts. To do so, we have converted the dataset of various font sizes into a standard size that is accepted by our model. The procedure by which we converted the dataset into a standard size, acceptable by our model is known as normalization. So, we have normalized our dataset at the preprocessing stage.

Building OCR model: To feed our dataset into the CNN layer, we have converted the dimensions of all the word segmented image data into 32×128. These words are then inputted into a CNN layer, which has a kernel size of (3, 3) with 64 nodes.

The output of the first CNN layer transferred to a pooling layer of kernel size (2, 2) which reduces the shape of the output to 16×64. The output of the previous pooling layer is forwarded to two CNN layers with 256 nodes on each layer. Each of these CNN layers is followed by a pooling layer. After these two CNN and pooling layers, the shape of the output data is reduced to 4×32. Again, this data is forwarded into two CNN layers with 512 nodes followed by two batch normalization layers. The output of the last batch normalization layer is transferred to one pooling layer and then to a CNN layer. In the last layer, we used an LSTM layer with 256 layers to recognize a word and finally saved it into a text file. This file is then passed to the answer evaluation part for grading.

4.2 Answer Evaluation

Data Preprocessing: In the data processing step, we removed all stop-words, punctuation, and any other special characters (if used) from the answers. Since the model was developed for short-question answer script evaluation, we kept the maximum sentence length at 40 for each answer. Therefore, if any answer is longer than the assigned length, we pruned them, and if smaller, we used *zero padding* to make them the same vector length.

Stemming: In NLP, stemming is a popular approach for text, word, or document normalization. Stemming shrinks inflection in words to their original forms. During application, a word is modified to communicate many grammatical phases, like tense, case, voice, person, number, gender, and mood. This versatility has an effect on NLP performance. To overcome this limitation, we used stemming, which reduces words to their basic form or stem.

Vector Representation of Answers: The embedding layer enables us to convert each word into a fixed length vector of a defined size. Word embedding is a word representation form that connects human perceptions of language to a machine. Embedding represents text in an N-dimensional space where words have the same meaning. The vector difference between similar words is very low and increases with a dissimilarity. The words are then represented by a vectors in a predefined vector space. Each word is mapped to one vector, and the vector values are learned in a way that resembles a neural network. After stemming, at this stage, we applied *one hot representation* on the corpus and finally developed an embedded matrix having $M \times N$ length. Here, M represents number of answers and N represents number of features.

Bidirectional LSTM Layers: Figure 4 shows the bidirectional LSTM architecture for anser-script evaluation. This architecture traverses the essay in both forward and backward directions. It then sums up the output of the two. We posited that a bidirectional LSTM would give us improved performance over a simple LSTM. We have considered each answer as a vector tokens and explored the use of bi-directional LSTMs for embedding tokens vector. In the case of bi-directional LSTMs, the two independent passes of the essay are then combined to predict the essay grade. These essay embeddings are then fed to a linear unit in the output layer which predicts the essay score. For the classification problem, we constructed a readout layer that predicted the probability of an answer having one of scores from one to three. For simplicity we applied one hot representation for the labels. Finally we used softmax

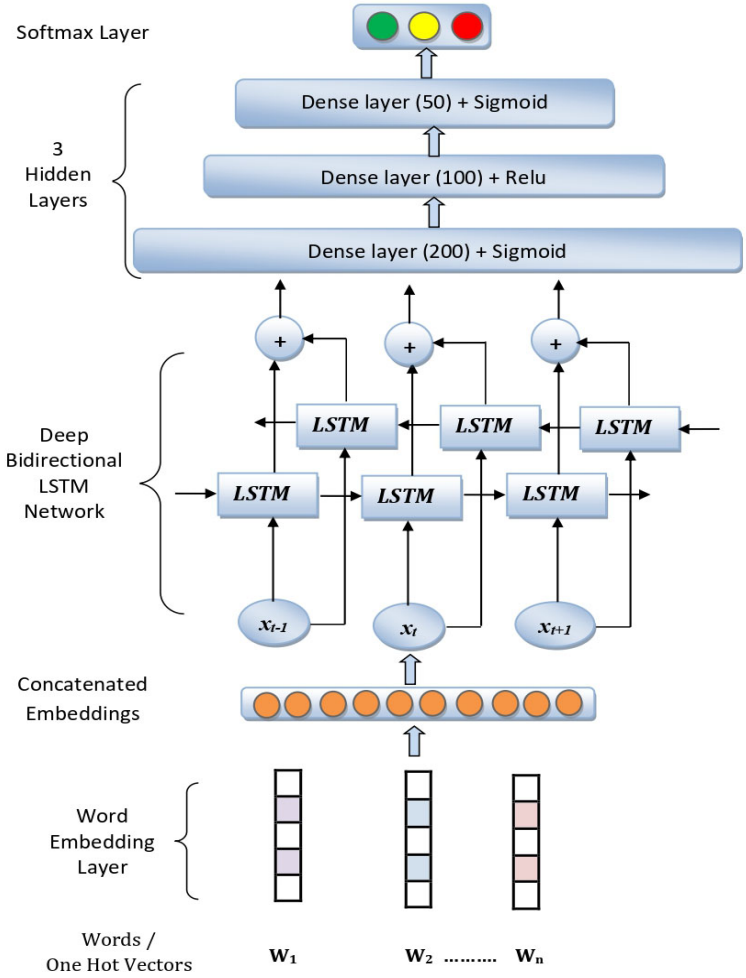


Figure 4: LSTM model for answer-script evaluation

activation function as follows:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K \tag{3}$$

Network Architecture: In our proposed sequential model, we assigned a maximum of 40 words to each answer. This indicates we need 40 time-steps and since the model is sequential for each answer, we picked the first word at time t_1, second at time t_2 and finally 40_{th} word at time t_{40}. We assigned a vocabulary size of 1000, therefore each word is represented in one hot vector of length 999. After applying pad sequence operation on the input, we found the vector with shape of (450,40) which is then passed to the embedding layer. This layer transformed the vector shape into

(450,40,40). The embedded length of 40, means each word is denoted by a vector of size 40. The vector is then passed into a bidirectional LSTM having 256 units. Since the model is bidirectional, the output shape will be (450,512). It means each answer having an embedded length of (40,40) is flattened into a (1,512) shape only. Since we used only 200 neurons for the first hidden layer, the output of BiLSTM is then flattened into the shape of (200,512). Followed by the same approach, the shape is then flattened into (200,100) dimensions in the second dense layer, (100,50) in the third dense layer, and finally (50,3) for the output layer. Since our proposed system has three categories of classes, we used categorical cross entropy to measure the distance between the predicted probability distribution and the actual probability distribution of the answers that can be defined as follows:

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i$$

where $\hat{y}_i$ is the i-th scalar value in the model output, y_i is the corresponding target value, and output size is the number of scalar values in the model output.

Hyperparameter Tuning The batch size is the number of training examples in one forward-backward pass. The higher the batch size, the more memory space it will need; the lower the batch size, the longer it will need. Depending on our GPU memory limit, we set the batch size to 16. Our three-layer-deep proposed network has more than 9.5 lac parameters. For non-linearity, we used a sigmoid activation function for the first and last layers and relu for the second layers. The learning rate of a model determines the speed of its weight update. Setting it too large makes the result exceed the optimal value, while setting it too small makes the falling speed too slow. In the training phase, we initialized the learning rate as 0.001 and assigned the learning rate as follows:

$$learning\ rate = initia\ lrate \times \frac{1}{(1 + (decay\ rate \times eppoc\ hnumber)} \tag{4}$$

A total of 50 epochs are used. We developed the model with different sizes of hidden layers, analyzed their effects, and picked the lightest one with excellent performance. To reduce overfitting, we also used dropout regularization technique. Depending on the depth and size of the recurrent models, the dropout ratio is different. For instance, when there are over three layers of LSTM and the hidden size is greater than 200, a dropout ratio is best set at 0.9; when there is only one layer of GRU cells and the hidden size is smaller than 100, the dropout ratio should be below 0.3. Therefore, considering the size of our developed model, we used a 0.3 dropout rate for the first hidden layer. The choice of optimization algorithm can make and adequate change to the results in deep Learning. Considering the problem's nature, we used adam optimzer.

5 Dataset and Performance Analysis

5.1 Dataset (for handwritten text recognition)

In order to convert the handwritten image text of the answer script into machine understandable text, we had to build an optical character reader (OCR) model. To train

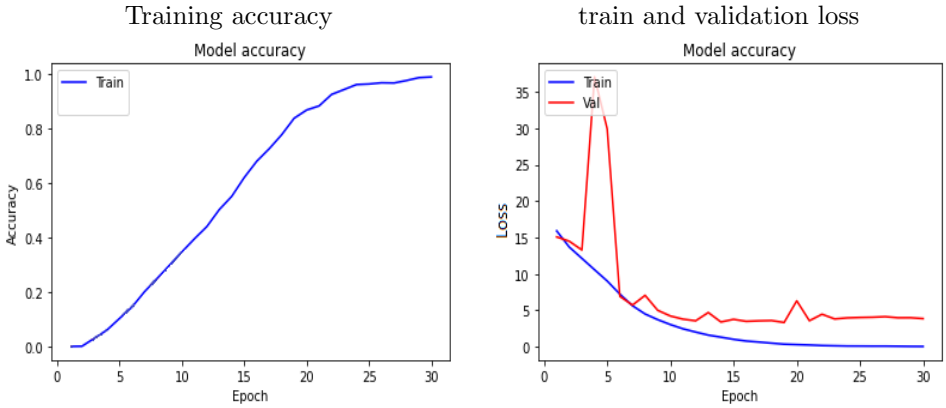


Figure 5: Performance of Handwritten text recognition system

the OCR model, we used IAM dataset which contains handwritten English sentences. This dataset consists of the Lancaster-Oslo-Bergen corpus, which is used for various recognition systems. Especially, when linguistic knowledge is given more priority than the lexicon level, this knowledge can be automatically obtained from the used corpus in the dataset. Some image-processing operations need to be done on the IAM dataset to segment it into lines and words. Then the segmented data is used to train our OCR model. We have also tested our OCR model to check its validity using random handwritten text data captured as an image using a camera. The training accuracy and the validation loss of the developed OCR model are shown in Figure 5.

5.2 Dataset (for answerscript evaluation)

The system is tested using a dataset of selected questions and answers from the students of the Computer Science and Engineering (CSE) department of Hamdard University Bangladesh. These answers were written as a part of mid-term exams in the introduction to information systems, artificial intelligence, and systems analysis and design courses.

Table 1 shows that for data-set development we prepared twelve basic questions, each of which has 35 to 40 answers at the undergraduate level of the CSE department. Since our developed model is for short answer script evaluation, we assigned the mark range from 1 to 3, indicating poor, average, and good respectively. We labeled the answers with the marks of a given range as per their correctness level. The higher marks indicate the greater correctness of an answer and vice versa.

5.3 Performance analysis

We used 450 answers as the dataset for the developed model; out of them 70% is used for training and 30% for testing purposes. Figure 6 (a) shows that our achieved training accuracy is just above 90% and the average test accuracy is nearly 80%. The train and test loss are also shown in Figure 6 (b), which says the loss is smoothly decreasing with the epochs. We also represented the outcomes of the model with a confusion matrix that visualizes the classification accuracy more clearly, as shown in

Table 1: The answer sets used in the experiment

Set No	Questions	No. of Answers
1	Define Computer.	40
2	Define prototyping. Why it's necessary?	40
3	What's the difference between data and information?	40
4	What's the relationships between hardware and software?	40
5	Which issues are considered when ANN is trained?	40
6	What's the role of a system analyst?	35
7	What is tangible and intangible cost/benefit?	35
8	What are the strategies of MIS planing?	35
9	How can we design an efficient AI system?	40
10	Write the Causes to terminate a project unsuccessfully.	35
11	Write the key issues to develop a system.	35
12	How alpha beta pruning improves search performance?	35
	Total Answers	**450**

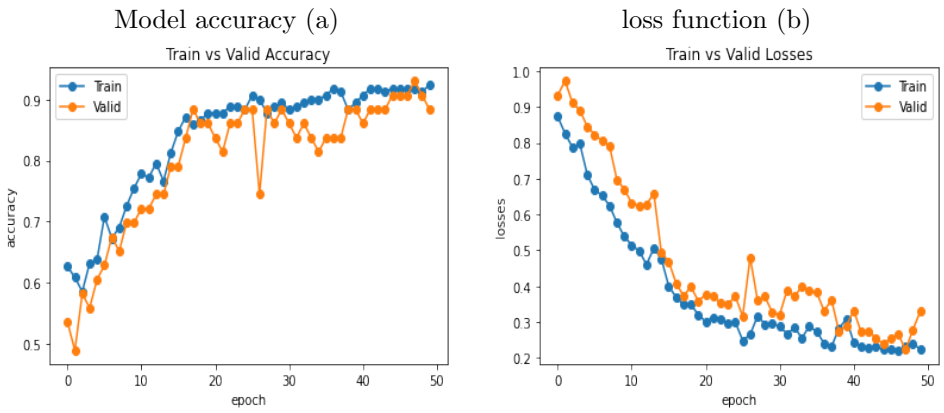


Figure 6: Performance analysis of Handwritten text grading system

Table 2: Precision, Recall and F1 score of proposed system

Grade	Precision	Recall	F1-score
lower	0.94	0.84	0.89
average	0.81	0.67	0.74
good	0.66	0.88	0.75

Figure 7. The figure says the true positive rate (TPR) and false positive rate (FPR) of classifying 'lower' grade is 94% and 6% respectively. The TPR of classifying 'average' grade is 82%; out of 43 average graded answers, our model mis-graded four answers to 'lower' and four answers to 'good' . Finally, for 'good' grade the TPR is 67% since of a variation of good answers. Around 31% good grades are classified into 'average' grades and 2% into 'lower' grade. We have also represented the result with precision, recall, and F1 score as mentioned in equation 5 to 7. The scores are shown in Table 2.

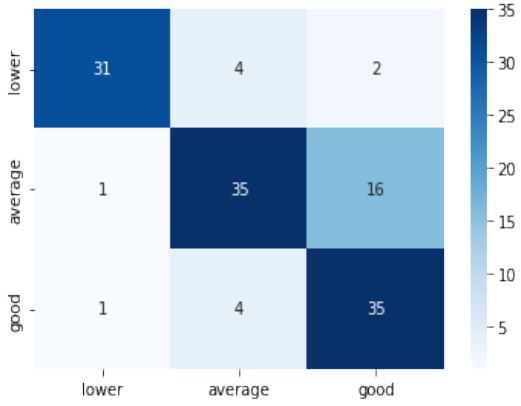


Figure 7: Confusion Matrix

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

6 Conclusion and Future Works

Developing a full-proof automated answer scrip evaluation system is really a big challenge since in an answer there may be a combination of figures, mathematical equations, and text with a variety of length, shape, and approach to the solution. In addition, evaluating handwritten answers of large size requires a complex network structure. In this paper, we developed a model just to evaluate answers of 40 words

long. The model is developed with a variety of possible approaches by adjusting parameters, deep layers, number of neurons, activation function, and bidirectional LSTM layers. We tuned each parameter many times and added or removed layers, LSTMs, or nodes to discover the lightest and optimal model. We also analyzed the performance of the model with the test set and achieved the nearest 80% accuracy and realized the accuracy could be improved with the enhancement of training data. This performance indicates that to develop a model for longer text (200-250 words) with figures and equations requires a higher level of analysis and study. We are working on the same with the aim of developing a model like expert human grader.

References

[1] Dong, F., Zhang, Y. (2016). *Automatic features for essay scoring–an empirical study.* In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (pp. 1072-1077).

[2] D. Alikaniotis, H. Yannakoudakis, and M. Rei, *Automatic Text Scoring Using Neural Networks*, arXiv:1606.04289v2 [cs.CL], 2016.

[3] B. Balci, D. Saadati, and D. Shiferaw, *Handwritten Text Recognition using Deep Learning*, Stanford University.

[4] T. Wang, D. Wu, A. Coates, A. Ng. "End-to-End Text Recognition with Convolutional Neural Networks" ICPR 2012

[5] Thodore Bluche, Jrme Louradour, Ronaldo Messina. *Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention.*, arXiv:1604.03286, 2016.

[6] A. Shehab, M. Elhoseny and A. E. Hassanien, "A hybrid scheme for Automated Essay Grading based on LVQ and NLP techniques," 2016 12th International Computer Engineering Conference (ICENCO), 2016, pp. 65-70, doi: 10.1109/ICENCO.2016.7856447.

[7] MitchelI, T., Russel, T., Broomhead, P., and Aldridge N., *Towards robust computerized marking of free-text responses*, In M. Danson (Ed.), Proceedings of the Sixth International Computer Assisted Assessment Conference, Loughboroug University, Loughborouh, UK., 2002.

[8] C. Cai, *Automatic Essay Scoring with Recurrent Neural Network*, Proceedings of the 3rd International Conference on High Performance Compilation, Computing and Communications, ACM, 2019.

[9] H. Nguyen and L. Dery, *Neural Networks for Automated Essay Grading*, Department of Computer Science, Stanford University.

[10] S. Srihari, J. Collins, R. Srihari, H. Srinivasan, S. Shetty, and J. B. Griffler, *Automatic scoring of short handwritten essays in reading comprehension tests*, Artificial Intelligence, Elsevier, 2008.

[11] A.A.A. Alil and S. Mallaiah, *Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout*, Journal of King Saud University – Computer and Information Sciences, Elsevier, 2021.

[12] A. Sharma and D. B. Jayagopi, *Automated grading of handwritten essays*, 16th International Conference on Frontiers in Handwriting Recognition, 2018.

[13] S. Ge and X.Chen, *The application of deep learning in automated essay evaluation*, Emerging technologies for education, Springer Nature, Switzerland, 310-318, 2020.

[14] P. Xu, T. M. Hospedales, Q. Yin, Y. Z. Song, T. Xiang, and L. Wang, *Deep Learning for Free-Hand Sketch: A Survey*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.

[15] Zhang, Haochen, Liu, Dong, Xiong, and Zhiwei, *CNN-based text image super-resolution tailored for OCR*, IEEE Visual Communications and Image Processing, 2017.

[16] C. Jin, B. He, K. Hui, L.Sun,*TDNN: A Two-stage Deep Neural Network for Prompt independent Automated Essay Scoring*, Association for Computational Linguistics, 2018

[17] P. Xu, C.K. Joshi, X. Bresson, *Multigraph Transformer for Free-Hand Sketch Recognition*, IEEE Transactions on Neural Networks and Learning Systems, 2021.

[18] Frinken V., Bunke H., *Continuous Handwritten Script Recognition*, In: Doermann D., Tombre K. (eds) Handbook of Document Image Processing and Recognition. Springer, 2014.

[19] Jorge Sueiras, Victoria Ruiz, Angel Sanchez, Jose F. Velez, *Offline continuous handwriting recognition using sequence to sequence neural networks*, Neurocomputing, Elsevier, Volume 289, 2018, Pages 119-128,

[20] Myat Thiri Wai, Thi Thi Zin, Mitsuhiro Yokota, Khin Than Mya, *Handwritten Character Segmentation in Tablet Based Application*, Proceedings of the 8th IEEE Global Conference on Consumer Electronics, Osaka, Japan, 2019.

[21] P. Shivakumara, D. Tang, M. Asadzadehkaljahi, T. Lu, U. Pal and M. Hossein Anisi, *CNN-RNN based Method for License Plate Recognition*, CAAI Transactions on Intelligence Technology, Vol. 3, No. 3, pp. 169-175, 2018.

[22] B. B. Klebanov and M. Flor, *Word association profiles and their use for automated scoring of essays*, In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 1148–1158.

[23] H. Bunke, M. Roth, E.G. Schukat-Talamazzini, *Offline Cursive Handwriting Recognition using Hidden Markov Models*, Elsevier Journal of Pattern Recognition, Volume 28, Issue 9, Pages 1399-1413.

[24] Thodore Bluche, Jrme Louradour, Ronaldo Messina. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention.

[25] Elfaik. H. and Nfaoui. E. H, *Deep Bidirectional LSTM Network Learning-Based Sentiment Analysis for Arabic Text*, Journal of Intelligent Systems, 2020.

[26] S. M. S. Islam, M. M. Hasan, S. Abdullah *Deep Learning based Early Detection and Grading of Diabetic Retinopathy Using Retinal Fundus Images*, arXiv:1812.10595v1 [cs.CV], 2018.

[27] Chen C-W, Tseng S-P, Kuan T-W, Wang J-F. *Outpatient Text Classification Using Attention-Based Bidirectional LSTM for Robot-Assisted Servicing in Hospital. Information. 2020.*

[28] I. K. Ihianle, A. O. Nwajana, S. H. Ebenuwa, R. I. Otuka, K. Owa and M. O. Orisatoki, *A Deep Learning Approach for Human Activities Recognition From Multimodal Sensing Devices*, in IEEE Access, vol. 8, pp. 179028-179038, 2020.

[29] Khan, M., Wang, H., Riaz, A. et al. *Bidirectional LSTM-RNN-based hybrid deep learning frameworks for univariate time series classification.* The Journal of Supercomputing 77, 7021–7045 (2021).

[30] Attali and Burstein *Automated essay scoring with e-Rater*, Journal of Technology, Learning, and Assessment, 4(3):1–30, 2006.

[31] Noura Farra, Swapna Somasundaran, and Jill Burstein, *Scoring persuasive essays using opinions and their targets.* In Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications, pages 64–74, 2015.

[32] Taghipour, K., Ng, H. T. *A neural approach to automated essay scoring.* In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (pp. 1882-1891), 2016.

[33] Chung, J., Gulcehre, C., Cho, K., Bengio, Y. *Empirical evaluation of gated recurrent neural networks on sequence modeling.*, arXiv preprint arXiv:1412.3555,(2014).

[34] Zhang, H., Litman, D. (2018). Co-Attention Based Neural Network for Source-Dependent Essay Scoring. In Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications (pp. 399-409).

[35] A. De Sousa Neto, B. Bezerra, A. Toselli and E. Lima, "HTR-Flor: A Deep Learning System for Offline Handwritten Text Recognition", Proceedings of International Conference on Graphics, Patterns and Images, pp. 1-8, 2020.

[36] Q. Vo, S. Kim, H. Yang and G. Lee, "Text Line Segmentation using a Fully Convolutional Network in Handwritten Document Images", IET Image Processing, Vol. 12, No. 3, pp. 438-446, 2018.

[37] MitchelI, T., Russel, T., Broomhead, P., and Aldridge N. , "Towards robust computerized marking of free-text responses", In M. Danson (Ed.), Proceedings of the Sixth International Computer Assisted Assessment Conference, Loughboroug University, Loughborouh, UK., 2002.

[38] B. Shi, X. Bai and C. Yao, An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition, arXiv:1507.05717v1 [cs.CV] 21 Jul 2015