# Automated Scoring for Short Questions with Deep Learning

Submitted by

Michelle Vanessa

U1620071L

Project Supervisor

Assoc Prof. Hui Siu Cheung

School of Computer Science and Engineering

AY2019/2020

NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

**SCE19-0305**

# Automated Scoring for Short Questions with Deep Learning

Submitted in Partial Fulfilment of the Requirement for the Degree of Bachelor of Engineering (Computer Science)

Submitted by

Michelle Vanessa

U1620071L

Project Supervisor

Assoc Prof. Hui Siu Cheung

School of Computer Science and Engineering

AY2019/2020

## Abstract

This report proposes an improved approach of short answer questions scoring with deep learning, namely Siamese Bidirectional LSTM model with feature engineering. The project will conduct several experiments in order to improve the performance of the model. The scope of the questions is limited to Data Structures questions, and Mohler dataset is used in this experiment [1]. Eventually, the model will be deployed into a web server for data collection so that the model can be further trained and for self-practice purposes for computer science students.

## Acknowledgements

The author would like to express her gratitude to various individuals for their continuous assistance. This project would not have been possible without their support.

Firstly, the author would like to extend her deepest gratitude to Assoc Prof Hui Siu Cheung, the author's final year project supervisor, for his guidance, encouragement, and constructive feedbacks throughout the year; to Akhil and Hao Anran for their continuous help and support; and to her parents who have supported her throughout her journey in Nanyang Technological University Singapore. Without them, this project would not have been as successful.

Lastly, the author wishes to thank all her friends who have supported during her time in Nanyang Technological University Singapore and have been there for her when she needed encouragement. Without their faith in her, the author would not have been able to complete this project smoothly.

# Table of Contents

# List of Figures

# List of Equations

# List of Tables

# Chapter 1: Introduction

## 1.1 Background

One of the main tasks of an educator is assess the understanding of their students. It can be measured through various assessments, which requires the educator to manually evaluate and grade the students' responses. Time taken to do such activity depends on the open-endedness assessment since answers of open-ended questions have wider variety and are more subjective. Thus, multiple choice questions will take shorter time to score compared to essay questions. Scoring open-ended questions is susceptible to the grader's subjectivity since there is no right or wrong. Automated grading system would help speed up scoring process and would be able to objectively grade student responses, thus eliminating the disadvantages of manual human scoring.

## 1.2 Motivation

The education sector would benefit greatly from the automated essay scoring. Despite the fact that multiple choice questions scoring is now mostly done by machine, automated essay scoring is still not widely used even though it has been a research topic for some time now.

Most educational system still requires human to manually grade them, which consumes a lot of the educators' time. With the help of the automated essay scoring, the grading process would be shortened [2], and educators would be able to reduce the time they spend doing so, and hence, increasing educators' productivity on other activities. Moreover, the automated essay scoring has numerous advantages over manual scoring, such as objectivity and consistency.

## 1.3 Objectives

The objective of this project is to improve the existing short answer scoring approach, namely the Siamese Bidirectional LSTM model. The approach will be improved by fine-tuning the hyperparameters of the model and modifying the architecture of the model itself. However, the scope of the questions is limited only to Data Structures questions so that the answers could be more topic specific. It would later on be implemented in an educational setting, where it could help facilitate the scoring process of short-answered data structures related C programming questions, which, in this case, is to assist scoring of

CX1007 Data Structures course as a part of students’ self-practice and performance evaluation.

# Chapter 2: Literature Review

The essay grading problem has been addressed since years ago. One of the earliest approach to the problem is a method known as C-rater [3], which was developed by Educational Testing Service, also widely known as ETS, to measure students' understanding based on their responses to short-answer questions. The method compares the syntactical characteristics of a sentence to a collection of correct ones. However, it is ignoring the difference between passive and active voice, such as "you need two plants" and " two plants are needed".

More recent work has analysed the difference between corpus-based and knowledge-based measures of text similarity, and it was shown using Pearson's correlation coefficient that corpus-based measure (Lexical Semantic Analysis) performs the best among other measures. It also introduced new technique which is similar to pseudo-relevance feedback to address the problem where there is more than one correct answer [4].

Although some works use Pearson's correlation coefficient to compare student and reference answers, another experiment [5] showed that similarity measure using Cosine coefficient produce the best result. Cosine similarity measure has the highest accuracy rate compared to other measures, namely Jaccard coefficient and Dice coefficient.

One of the works that use Cosine similarity is the paragraph embeddings [6], which focused on short answer scoring. Answers are considered short "if its length approximately ranges from one phrase to one paragraph". The word embedding vectors from the answer are combined using average, sum, or other methods, then, using the calculated vectors, new vectors are generated using paragraph embedding model. Cosine coefficient will be used to compare the paragraph vectors.

Another work proposed a new technique named question demoting [1]. The technique removes any words that occur in the question from both the student and gold standard answers. This technique is implemented to eliminate the chance that students who repeat words from the question in their answers get high score as it does not reflect their understanding in the topic.

Given the numerous methods, such as text similarity, question demoting, term weighting, length ratio, and word embeddings, some may affect the accuracy more than others. It is proven that alignment has the most significant influence. It measures text semantic similarity using text's lexical and contextual meaning, which are represented by a word aligner. The rooted mean squared error (RMSE) increased by 5.7% when alignment is removed from the method. On the other hand, question demoting, length ratio, and term weighting improve the RMSE by 1.8%, 0.1%, and 0.2%, respectively [7].

RNNs are designed to retain information from previous time frames so that the patterns found in the past information can be used to predict the future patterns. Such method is known as long-term dependencies, and RNNs are designed to handle that. However, in practice, RNNs are unable to learn the dependencies, so a new approach, Long Short-Term Memory, was introduced to address this issue [8].

Long Short-Term Memory unit is a type of Recurrent Neural Network. the information in this unit goes back in time. It was first introduced in 1997 due to the inability of the conventional approach at that time to prevent the information going backwards to blow up or vanish [9]. To avoid the aforementioned problem, a forget gate was added to the unit so that insignificant information can be discarded.

With the discovery of more complex deep learning approaches, the demand of more advanced neural network training techniques became higher. Motivated by the uneven distribution of neural network layers' input during training, batch normalization was introduced in 2015 [10]. The technique is designed to balance the distribution of the inputs by reducing internal covariate shift (ICS), which is the change in the distribution of the input variables in training and test data [11].

Despite the breakthrough of machine learning techniques, overfitting is still a prominent issue in deep learning, especially when the size of training set is very small compared to the complexity of the network. In this case, dropout can be used to avoid the problem. The concept of dropout is to randomly drop neurons from the network, so that dropped neurons will be omitted during training, and hence reducing the complexity of the network [12]. Therefore, the network would be able to learn information that is significant to produce the correct output [13].

# Chapter 3: Existing Approach

The existing approach automates grading of short answer using Siamese bidirectional LSTM-based regression. This chapter discusses the deep learning methods utilised in the approach.

## 3.1 Architecture

This approach implements several neural network architectures which are combined, and together, they build the overall model architecture. The model consists of several layers, which are shown in  Figure 1, the illustration of the layers in the network.

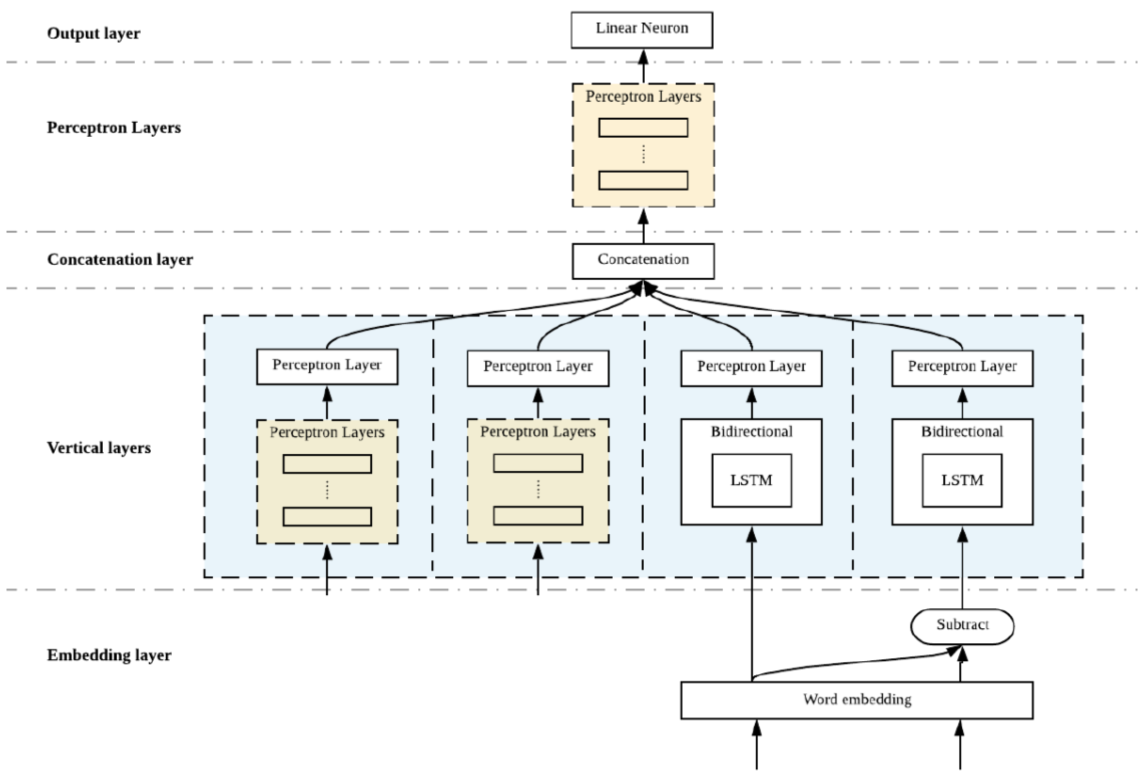


*Figure 1 Network architecture illustration*

Moreover, Figure 2 below is the neural network graph of the model as generated by the visualization tool, TensorBoard.

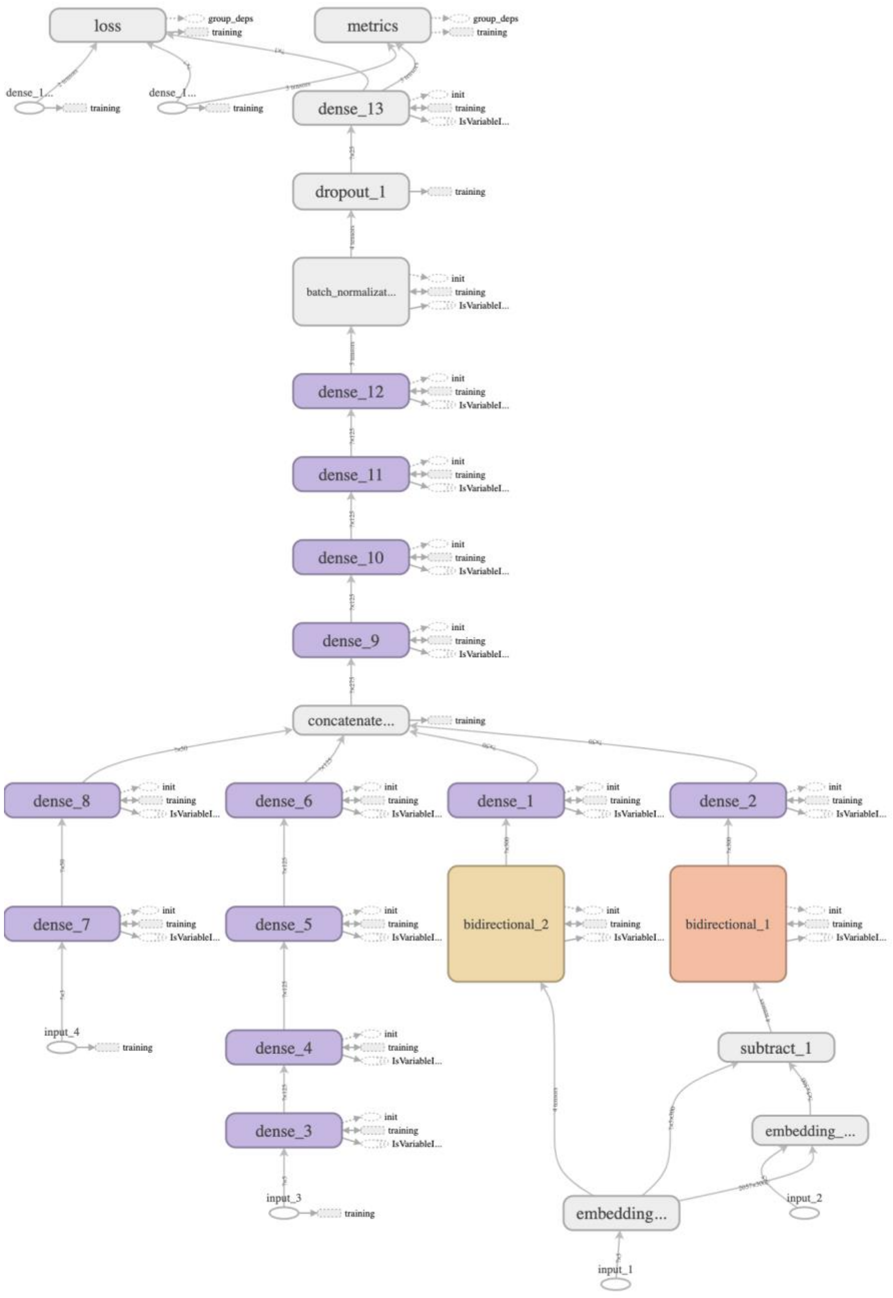


*Figure 2 Neural network graph generated by TensorBoard*

### 3.1.1 Embedding Layer

The first layer of the network is the embedding layer. This layer takes in a sequence of words as input, and each word will be mapped into high-dimensional vectors representing the respective words, also known as word embedding. In word embedding, similar words have similar values, and vector operations can be performed to obtain other words' vector values.

This approach uses Word2Vec, which is one of the word embedding models. The model utilises the Skip-gram model and negative sampling [14]. The words are represented in such a way that the result of the vector operations reflects the linguistic patterns of the words. For example, the operation of the vectors “Madrid” – “Spain” + “France” will result a vector close to the vector of “Paris” [15].

The input of this layer will be the student answers and the reference answers. On the other hand, the output of this layer will be the mapped 300-dimensional word vectors of the respective answers.

### 3.1.2 Vertical Layers

This layer consists of several vertical layers that is independent from each other, and each has its specific functions. Following are the details of the layers as shown in Figure 2 from left to right, respectively.

#### 3.1.2.1 Multilayer perceptron (feature engineering)

This layer consists of 2 layers of neurons, both of 50 neurons with sigmoid activation function. It receives 3 integers. The first integer is number of words of the reference answer, the second is number of the student answer, and the last one is number of words that exist both in the student and reference answers. To obtain these numbers, the answers are first tokenized using NLTK word tokenizer.

#### 3.1.2.2 Multilayer perceptron (feature engineering)

This layer takes input of 5 set of data obtained from feature engineering. Feature engineering is a process where a certain data is produced by transforming raw data so that it could be used by the model to better learn the pattern of the given data [16]. In this case, length of the student answer (number of characters), ratio of the length of the reference answer and length of the student answer, number of words in the student answer, and the number of unique words in the student answer are used.

The input is then processed through 4 layers of 125 neurons with sigmoid activation function.

#### 3.1.2.3 LSTM unit (reference answer)

This layer receives input from embedding layer, which is the embedded student answers. It will then be propagated through a bidirectional LSTM unit and through a layer of 50 neurons with sigmoid activation function.

3.1.2.4 LSTM unit (comparison with student answer)

This layer architecture is the same as the previous layer. However, it has different input. Instead of receiving input straight from the embedding layer, this layer receives a vector of comparison between student answer and reference answer. The two sentences are compared using the distance of the two vectors. The distance is calculated using the equation below, where $R_i$ is the reference answer of question $Q_i$, and $A_{ij}$ is the student answer j of question $Q_i$. *v(X)* is the word vector of sentence *X*.

$$dist = v(R_i) - v(A_{ij})$$

*Equation 1 Distance between two vectors*

Word2Vec groups similar words together [15], so words with similar meaning has similar vector values. Hence, the distance of the two vectors should be small if the sentences have similar meaning.

### 3.1.3 Concatenation Layer

On this layer, the output from vertical layers are merged into one vector to be processed further in the following layers.

### 3.1.4 Perceptron Layers

The concatenated vector is then propagated through a multilayer perceptron. The multilayer perceptron has 4 layers, where the first 3 layers consist of 125 neurons, and the last one consists of 25 neurons.

### 3.1.5 Output Layer

The final layer of the network consists of 1 neuron with linear activation function. This neuron outputs the final predicted score.

## 3.2 Deep Learning Models

There are two main neural network architectures that are used in this approach, namely multilayer perceptron and bidirectional LSTM unit.

### 3.2.1 Multilayer perceptron

Multilayer perceptron, a class of feedforward neural network, is a deep learning model that approximates some function by learning parameter so that it could generate the best result. Information flows through the layers of neurons, the first one, being called first layer, until the last one, the output layer. Between the first and output layer, there are more layers, called hidden layers. In contrast to Recurrent Neural Network,  information in multilayer perceptron never goes backwards [17].

In this approach, multilayer perceptron is combined with other model to form the overall architecture. The output of the perceptron is concatenated with the output from other models before going through another chain of multilayer perceptron.

### 3.2.2 Bidirectional LSTM Unit

The basic LSTM unit consists of a cell state that modulates information through the unit, and three gates (forget, input, and output) [8]. Following is the functions used in the unit, where $x(t)$ is the input and $h(t)$ the output.

#### 3.2.2.1 Forget gate

Forget gate will determine whether the information should be removed or not. When it is 0, nothing will go through.

$$f(t) = \sigma(U_f^T x(t) + W_i^T h(t-1) + b_f)$$

*Equation 2 Forget gate*

Where σ is sigmoid function.

#### 3.2.2.2 Input gate

##### 3.2.2.2.1 Sigmoid layer

This layer will determine which values to update.

$$i(t) = \sigma(U_i^T x(t) + W_i^T h(t-1) + b_i)$$

*Equation 3 Input gate sigmoid layer*

##### 3.2.2.2.2 Tanh layer

This layer will produce vector of new candidate values.

$$\tilde{C}(t) = \phi(U_c^T x(t) + W_c^T h(t-1) + b_c)$$

Equation 4 Input gate tanh layer

Where ϕ is tanh function.

### 3.2.2.3 Output gate

This gate will allow memory cell to have effect on other neurons.

$$o(t) = \sigma(U_o^T x(t) + W_o^T h(t-1) + b_o)$$

Equation 5 Output gate

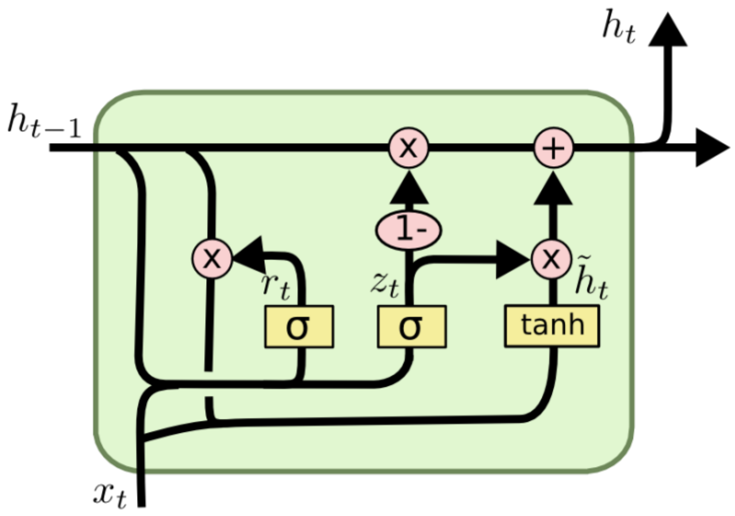


Figure 3 LSTM Unit

Afterwards, the cell state is updated using equation below, where ⊙ is element-wise product.

$$C(t) = \tilde{C}(t) \odot i(t) + C(t-1) \odot f(t)$$

Equation 6 Cell state

The output value is calculated using the equation below.

$$h(t) = \phi(C(t) \odot o(t))$$

Equation 7 LSTM output

However, this approach uses bidirectional LSTM, which means the neurons are split into two directions, forward and backward. This method will enable the effective usage of both past and present information for a specific time frame [18].

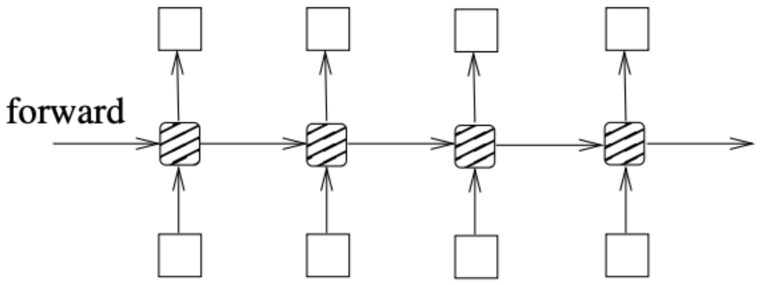


*Figure 4 LSTM network*

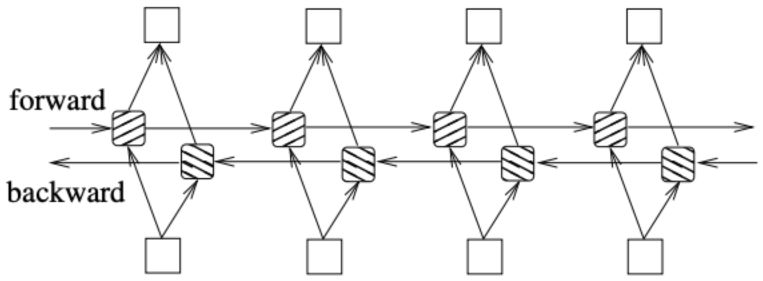


*Figure 5 Bidirectional LSTM network*

Two bidirectional LSTM units with identical architecture are used for this approach. Hence, this model is also called a Siamese Bidirectional LSTM.

## 3.3 Techniques

### 3.3.1 Batch Normalization

The input and output of batch normalization layer are four dimensional vectors of batch, channel, and two spatial dimensions [19]. It normalises each input of a layer by subtracting mini-batch mean and dividing it by the mini-batch standard deviation, but it may change the input's representation [11].

For input $x = \{x_{1\ldots m}\}$ and output $y_i$ over a mini-batch B, batch normalization needs to learn parameters $\gamma$ and $\beta$, where $\epsilon$ is a constant added for numerical stability.

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$$

*Equation 8 Mini-batch mean*

$$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

*Equation 9 Mini-batch variance*

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

*Equation 10 Normalization*

$$y_i \leftarrow \gamma \hat{x}_i + \beta$$

*Equation 11 Scaling and shifting*

### 3.3.2 Dropout

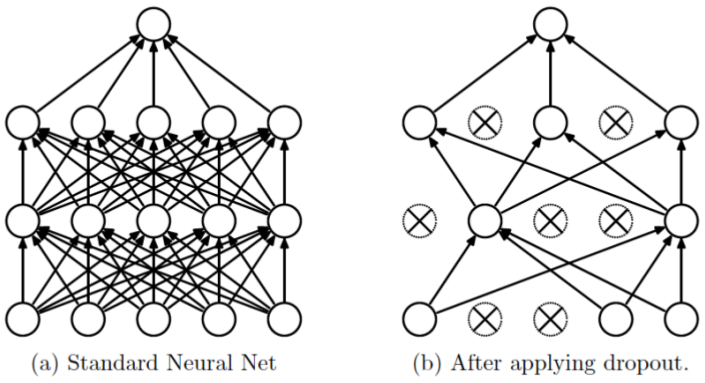


*Figure 6 Dropout on neural network with 2 hidden layers*

In this model, we use dropout with parameter 0.5, which means that 50% of the neurons, randomly selected, are dropped or omitted during training.

# Chapter 4: Experimental Evaluation

## 4.1 Dataset

The experimental evaluation uses a dataset from an experiment conducted by Mohler, Bunescu, and Mihalcea [1].

The dataset consists of data structures questions for introductory computer science assignment at the University of North Texas. There are total of 80 questions and 31 students enrolled in the course. In total, the dataset consists of total 2273 student answers since some students did not answer some questions.

Each answer is graded manually by two human graders. The score is an integer ranging from 0, for incorrect answer, to 5, for perfect answer. The average of the two scores is then used as gold standard of this experiment.

## 4.2 Evaluation Metrics

### 4.2.1 Pearson Correlation Coefficient

Pearson correlation coefficient measures the linear correlation of two variables. The coefficient ranges from -1 to 1, where 1 means the two variables are positively correlated and -1 negatively correlated.

The Pearson correlation coefficient $\rho$ of two variables $X$ and $Y$ is defined as below.

$$\rho_{X,Y} = \frac{s_{X,Y}}{s_X s_Y}$$

*Equation 12 Pearson correlation coefficient*

$$s_{X,Y} = \frac{1}{n-1}\sum_{k=1}^{n}(X_k - \overline{X})(Y_k - \overline{Y})$$

*Equation 13 Covariance*

$$s_X = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(X_k - \overline{X})^2}$$

*Equation 14 Standard deviation*

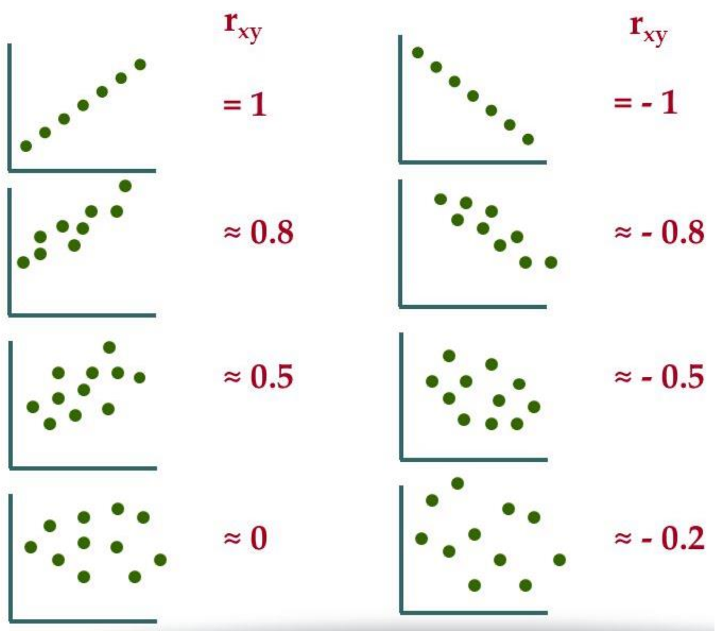


*Figure 7 Pearson correlation coefficient*

Pearson correlation coefficient is calculated to evaluate the approach. However, it only represents the linear correlation of the two vectors and does not represent how much the vector deviate from each other. For example, vectors [3,4,5] and [1,2,3] will have the coefficient equals to 1. Therefore, this metric cannot be used alone to determine the performance of a model.

### 4.2.2 Rooted Mean Square Error

Rooted mean square (RMSE) is the square root of the arithmetic mean of the squares of the difference between the two variables.

$$RMSE = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(y_k - \hat{y}_k)^2}$$

*Equation 15 RMSE*

Where $y_{1\ldots k}$ are observed values and $\hat{y}_k$ are predicted values. Since this metric represents the difference between the two values, this metric can be used to observe the performance of a model. Lower RMSE means the model could predict values that are closer to the given dataset.

### 4.2.3 Mean Absolute Error

Mean absolute error (MAE) measures the difference of two variables. It is the arithmetic mean of the absolute difference of the two variables.

$$MAE = \frac{1}{n-1}\sum_{k=1}^{n}|y_k - \hat{y}_k|$$

*Equation 16 MAE*

Where $y_{1\ldots k}$ are observed values and $\hat{y}_k$ are predicted values. This metric also represents the difference between the two values and can be used to evaluate model performance. Lower MAE also means that the model could predict values close to the actual data.

## 4.3 Data Augmentation

The dataset only consists of 2273 tuples, which is not big enough to train the model. This number of data is not sufficient to train a complex network, and it will cause the model to overfit the data. Hence, to avoid this issue, data augmentation is applied to obtain larger dataset.

Data augmentation is a method to increase the size and diversity of the dataset for training without actually collecting new data, so the new data is obtained from the dataset itself [20].

In this experiment, student answers that are given perfect score 5.0 by human graders are used as new reference answer. Thus, if $n$ students received perfect grade in a particular question, $(n-1) * m$ new data tuples can be generated for that question. By performing this technique, around 35000 data samples were generated and used to train the model.

## 4.4 Evaluation

### 4.4.1 Implementation

The model was built on Python using Keras, a deep learning library that works on top of Tensorflow.

### 4.4.2 Model Selection

The model selection method used was a combination of three-way data splits and 5-fold cross validation methods.

The dataset was first split into training and test set on 80:20 ratio. Then, the model is trained using 5-fold cross validation of the training data, and the best model is chosen based on the average of RMSE and MAE. The fold that has lowest average of RMSE and MAE is selected. Lastly, the chosen model is used to predict values from unseen data (test set) and its result is used to evaluate the overall model performance.

The data split ratio 80:20 is chosen upon some considerations. The training set, which is used to train the model, might be too small if 70:30 ratio is used, considering the complexity of the model. On the other hand, 90:10 ratio will produce very small test set, which is used to assess the performance of the chosen model. If the test set is too small, it may not correctly represent the model performance.

#### 4.4.3 Training

For each fold on training set, the model is trained on GPU using 150 epochs since the loss seems to be stable around that iteration. The parameter is updated using mini-batch stochastic gradient descent algorithm, specifically AdaGrad optimization method, and the loss function is mean absolute error (MAE) as stated in Equation 16.

AdaGrad is used because it is not sensitive to hyperparameters as it is dynamic and adaptable to the data and generate different learning rates for different features [21]. The input of this model is diverse, and hence, using AdaGrad as the optimization method will help the model to train better.

## 4.5 Performance of Existing Model

Using the evaluation techniques explained in the previous section, the existing model performance was as stated below.

| **Pearson coefficient** | **RMSE** | **MAE** | **Avg(RMSE+MAE)** |
|---|---|---|---|
| 0.4416 | 0.9815 | 0.6471 | 0.8143 |

*Table 1 Performance metrics of existing approach*

From the metrics above, it is shown that the model does not perform very well as the Pearson correlation coefficient is less than 0.5. The low coefficient means that the predicted values are not linearly correlated to the target values.

Moreover, the MAE is also quite high. It means that for all gold standard score, the average difference with the predicted score is 0.6471 points, which is 12.94% of the score. For example, if the gold standard score is 4, the model would probably predict the score to be 4.6471, which is closer to 5.

Below is the training and validation loss against the number of iteration.

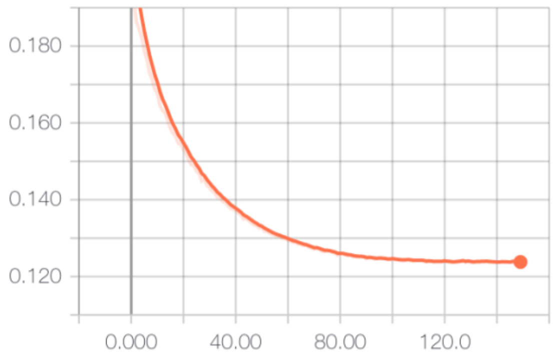


*Figure 8 Training loss of original model*

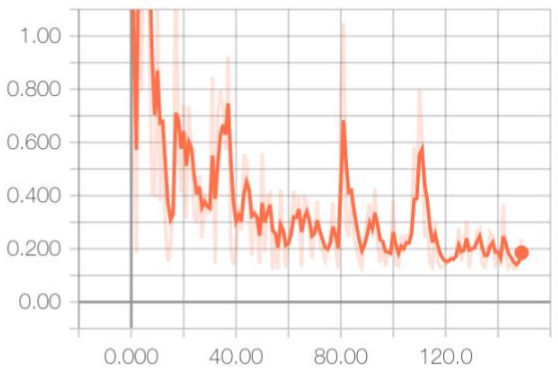


*Figure 9 Validation loss off original model*

As we can see from the graphs above, the graph of the validation loss over number of iteration seems to be very spiky even though the model seems to perform well on training data. It means that the model does not perform well on the unseen data and is overfitting on the training dataset.

## 4.6 Improving Model Performance

Since the original model is overfitting on the training data, the model should be improved to result on better performance. There are many techniques that can be implemented to improve the model performance. Such techniques will be discussed further in the following sections.

### 4.6.1 Recurrent Batch Normalization

The original model only perform batch normalization on the perceptron layer. In this experiment, additional batch normalization is applied on each of the LSTM layers.

### 4.6.2 Regularization

Regularization is an strategy to reduce error on unseen data or test error by altering the learning algorithm [17]. This is done by adding penalty or regularization term on the cost or loss function. Even though this could increase the training error, reducing test error would improve the overall performance of a model as it shows that the model is not overfitting on the training data.

There are some regularization techniques that are implemented in this experiment.

#### 4.6.2.1 Weight regularization

The most commonly used weight regularization is $L_2$ regularization, also known as ridge penalization. This approach uses $L_2$ regularization term, defined as below [22].

$$L_2\ regularization\ term = \ w_1^2 + w_2^2 + \cdots + w_n^2$$

*Equation 17 $L_2$ regularization term*

This term is added to the cost function $J$. Since the goal of the algorithm is to minimize the cost function, when the penalization $\beta_2$ is big, the weights $w_{ij}$ would be small.

$$J \leftarrow J + \beta_2 \sum_{i,j} (w_{ij})^2$$

*Equation 18 Cost function with $L_2$ regularization*

In this experiment, different values of parameter $\beta_2$ will be used to evaluate the optimal parameter.

#### 4.6.2.2 Dropout

As explained in Section 3.3.2, dropout is a method to avoid overfitting by omitting some neurons in the network. In the original model, the neurons in perceptron layer is dropped out 50%. Hence, this experiment will try to perform dropout on the LSTM layer and try different dropout rates on the perceptron layer.

### 4.6.3 Question Demoting

During data pre-processing, all words in reference and student answers that are also in the question are removed. However, there are some cases where the question contains its answer. For example, optional questions like “What data structure is more appropriate for scheduling printing jobs at a printer, a stack or a queue?” will have answer either “Stack” or “Queue”. When question demoting is performed on this question, the answer will be empty. Hence, the options, the phrase “a stack or a queue” is removed from the question.

### 4.6.4 Classification Model

The model is converted into a classification model with expectation that it will improve the performance.

#### *4.6.4.1 Motivation*

The answers from the dataset is scored using ordinal values, which are 0, 1, 2, 3, 4, and 5. However, there is no rubric on how to score the answers, so each grader may have different opinion on how an answer should be graded. For example, one might think an answer should be graded 2, while the other think it is 4.

Moreover, the model that is used to predict the score is a regression model. The predicted answer would be a continuous number, and since there is no rubric on how to score the answer, there is no way to determine how correct an answer is based on the score.

#### *4.6.4.2 Implementation*

The model is modified into a multi-category classification model, where the output is three classes, 0, 1, and 2. An answer is labelled 0 or ‘wrong’ when it is completely wrong and completely different from the reference answer. It is labelled 1 or ‘not quite correct’ when it is only partially correct. Lastly, it is labelled 2 or ‘correct’ if it is a perfect answer and exactly the same as the reference answer.

The model is built by changing the output layer of the original model to a softmax layer with 3 outputs. Then, it is trained with the dataset, where an additional pre-processing is applied. The target output, which is the average scores of two graders $x$, are discretized into 3 categories using the function below.

$$f(x) = \begin{cases} 0, & x \leq 1 \\ 1, & 1 < x < 4 \\ 2, & x \geq 4 \end{cases}$$

*Equation 19 Target output discretization*

### 4.6.4.3 Evaluation metrics

Accuracy, precision, recall, and F1 score are used to evaluate the model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is the percentage of data that are correctly predicted by the model.

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

Precision is the ratio of data of a class that is correctly predicted over the size of data that is predicted to be in that class, and it is biased by the false positive

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

Recall is the ratio of data of a class that is correctly predicted over the size of data that is actually in that class, and it is biased towards the false negative.

$$F1\ score = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

Since the three measurements are biased, F1 score, which is biased towards false positive and false negative was introduced.

## 4.7 Optimization Results

This section will discuss the performance of the model when improvement techniques from the previous section are applied. For each technique, different values of parameter, if applicable, are implemented and evaluated to obtain the best result. Note that the most optimal values are in bold.

### 4.7.1 Recurrent Batch Normalization Results

| Model | Pearson | RMSE | MAE | Avg(RMSE,MAE) |
|---|---|---|---|---|
| Original | 0.4416 | 0.9815 | 0.6471 | 0.8143 |
| Recurrent BN | **0.4426** | **0.9648** | **0.6429** | **0.8039** |

*Table 2 Experiment results on recurrent batch normalization*

By applying batch normalization on the recurrent network, the model performance is improved for all metrics.

### 4.7.2 Regularization Results

Several experiments on regularization were conducted on different part of the model. Different values of parameter are also experimented to obtain the best result of the regularization.

#### *4.7.2.1 Weight regularization on LSTM units*

Experiments on different parameters of the $L_2$ regularization is conducted. All experiments have same parameter value for bias, weights, and recurrent parameters.

| Parameter | Pearson | RMSE | MAE | Avg(RMSE,MAE) |
|---|---|---|---|---|
| 0.01 | 0.4512 | 0.9787 | 0.6293 | 0.8040 |
| **0.001** | **0.4526** | **0.9556** | **0.6466** | **0.8011** |
| 0.0001 | 0.4513 | 0.9563 | 0.6467 | 0.8015 |

*Table 3 Experiment results of weight decay on LSTM units*

The result shows that parameter 0.001 has the best performance among other parameters on all metrics.

#### *4.7.2.2 Weight regularization on perceptron*

Weight decay was applied to the last 2 perceptron layers on the 2 multi-layer perceptron on vertical layer and perceptron layer (refer to Figure 1).

Values for bias and weights parameter are the same, as stated below.

| **Parameter** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| **0.01** | 0.4471 | **0.9694** | **0.6309** | **0.8002** |
| 0.001 | 0.4491 | 0.9677 | 0.6571 | 0.8124 |
| 0.0001 | **0.4504** | 0.9952 | 0.6349 | 0.8151 |

*Table 4 Experiment results of weight decay on perceptron*

Weight decay with bias and weight parameter 0.01 results on model with the best performance. Even though parameter 0.0001 has the highest Pearson correlated coefficient, as elaborated on Section 4.2.1, it cannot be used independently to evaluate model performance.

#### *4.7.2.3 Dropout on LSTM*

Dropout with different rates are applied to the model on the LSTM layers.

| **Rate** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| 0.2 | 0.4476 | 0.9697 | 0.6445 | 0.8071 |
| 0.4 | **0.4506** | **0.9632** | **0.6432** | **0.8032** |
| 0.5 | 0.4413 | 0.9599 | 0.6629 | 0.8114 |

*Table 5 Experiment results of dropout on LSTM layers*

From the experiment, it is found that dropout rate 0.4 has the best result among other rates.

#### *4.7.2.4 Dropout on perceptron*

On the original model, dropout rate 0.5 is used on the perceptron layers. To find out the optimal rate, first lower rate is used to see if it improves the model. Then, after seeing that lower rate does not improve the performance, higher rate is used.

| **Rate** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| 0.2 | 0.4436 | 1.0034 | **0.6369** | 0.8202 |
| 0.5 | 0.4416 | 0.9815 | 0.6471 | 0.8143 |
| 0.6 | **0.4504** | **0.9778** | 0.6450 | **0.8114** |

*Table 6 Experiment results of dropout on perceptron layers*

Dropout with rate 0.6 has the best overall performance even though its MAE is still higher than model with rate 0.2.

All regularization methods are combined, and based on the experiments above, the optimal parameter values are used.

| **Model** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| Original | 0.4416 | **0.9815** | 0.6471 | **0.8143** |
| Regularized | **0.4486** | 1.0199 | **0.6294** | 0.8247 |

*Table 7 Experiment results on regularization methods*

However, the performance decrease as the model seem to be underfitting the data. Hence, dropout rate is reduced back to 0.5, and afterwards, recurrent batch normalization is implemented along with all the regularization methods.

| **Model** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| Original | 0.4416 | 0.9815 | 0.6471 | **0.8143** |
| Regularization methods | **0.4494** | **0.9504** | 0.6523 | **0.8014** |
| Regularization and recurrent batch normalization | 0.4461 | 1.0000 | **0.6200** | 0.8100 |

*Table 8 Experiment results on regularizations*

The overall performance is better after the dropout rate is reduced. Then, recurrent batch normalization is added to the model. However, the performance does not seem to be improved.

### 4.7.3 Question Demoting

Question demoting is applied to the dataset, and the original model is trained using the demoted answers.

| **Model** | **Pearson** | **RMSE** | **MAE** | **Avg(RMSE,MAE)** |
|---|---|---|---|---|
| Original | 0.4416 | 0.9815 | 0.6471 | 0.8143 |
| Question demoting | **0.469** | **0.9761** | **0.6204** | **0.7983** |

*Table 9 Experiment results on question demoting*

The performance improved quite significantly after question demoting is applied. Hence, question demoting is performed along with the optimal models from the previous section.

| Model | Pearson | RMSE | MAE | Avg(RMSE,MAE) |
|---|---|---|---|---|
| Regularized | **0.4639** | **0.9604** | 0.6124 | **0.7864** |
| Regularized and BN | 0.4601 | 0.9816 | **0.6091** | 0.7954 |

*Table 10 Experiment results on optimal model with question demoting*

The model without recurrent batch normalization is shown to perform better as the overall metrics are better, and among other improvement, this method has the highest performance.

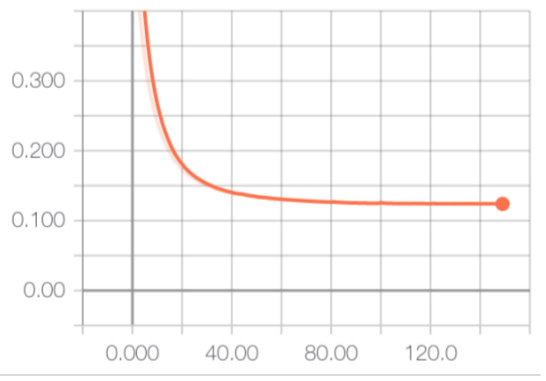


*Figure 10 Training loss of optimal model*

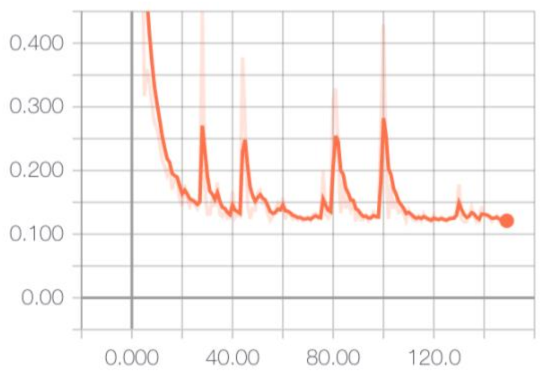


*Figure 11 Validation loss of optimal model*

Compared to the original model, the performance of the model after improvement is better, especially on unseen test data. The validation loss is significantly lower and more stable than the one from the original model.

### 4.7.4 Classification Results

Using the parameters from the previous sections, the classification model is trained using dataset that has been applied question demoting.

#### *4.7.4.1 Original model*

| Class | Precision | Recall | F1 Score | Support |
|---|---|---|---|---|
| 0 | 0.00% | 0.00% | 0.00% | 92 |
| 1 | 55.53% | 21.01% | 30.49% | 1266 |
| 2 | 80.68% | 95.68% | 87.54% | 4726 |
| **Weighted avg** | 74.23% | 78.70% | 74.35% | |
| **Accuracy** | 78.70% | | | |

*Table 11 Experiment results on classification model*

#### *4.7.4.2 Original model with question demoting*

| Class | Precision | Recall | F1 Score | Support |
|---|---|---|---|---|
| 0 | 31.13% | 51.09% | 38.68% | 92 |
| 1 | 51.12% | 36.05% | 38.32% | 1266 |
| 2 | 83.09% | 90.96% | 86.85% | 4726 |
| **Weighted avg** | 75.65% | 77.81% | 76.02% | |
| **Accuracy** | 77.81% | | | |

*Table 12 Experiment results on classification model with question demoting*

#### *4.7.4.3 Model with regularization and question demoting*

| Class | Precision | Recall | F1 Score | Support |
|---|---|---|---|---|
| 0 | 0.00% | 0.00% | 0.00% | 92 |
| 1 | 47.96% | 19.51% | 27.73% | 1266 |
| 2 | 80.97% | 95.41% | 87.60% | 4726 |
| **Weighted avg** | 72.87% | 78.17% | 73.82% | |
| **Accuracy** | 78.17% | | | |

*Table 13 Experiment results on classification model with regularization and question demoting*

Based on the results above, it can be concluded that applying regularization and question demoting does not improve the performance of the model as the accuracy is lower.

# Chapter 5: System Implementation

## 5.1 Software Architectural Patterns

The system is implemented using Django framework, and its interface is built using React library.

Django is an open-source, high-level Python web framework that supports rapid development and clean, pragmatic design [23]. It follows the Model-View-Template architecture pattern, also known as MVT. The abstraction layer, model, provides structuring and manipulating the data. The view layer is responsible for the logical operations and processing user requests, including returning the responses. Lastly, template layer renders the information to be presented to the user [24].

In this system, the template layer is implemented with the support of Django REST Framework and React library. Django REST Framework, which stands for Representational State Transfer, is a powerful and flexible toolkit for building Web APIs [25]. The API helps ease the usage of Django Server as an REST API. Information from Django Server is passed through Django REST Framework to React App. React App will then provide interface that users can see.

## 5.2 System Interface

### 5.2.1 General

#### *5.2.1.1 Sign up*

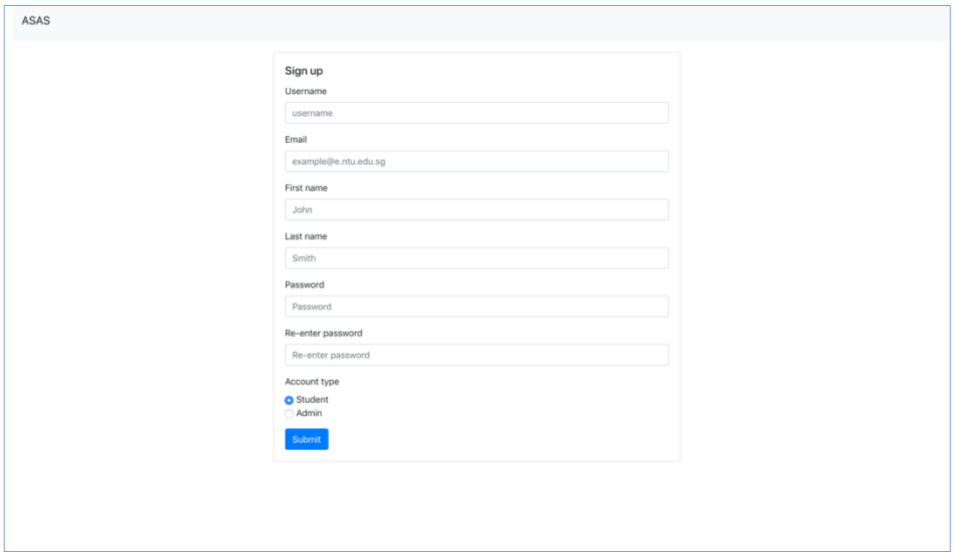


*Figure 12 Sign up page*

#### *5.2.1.2 Log in*

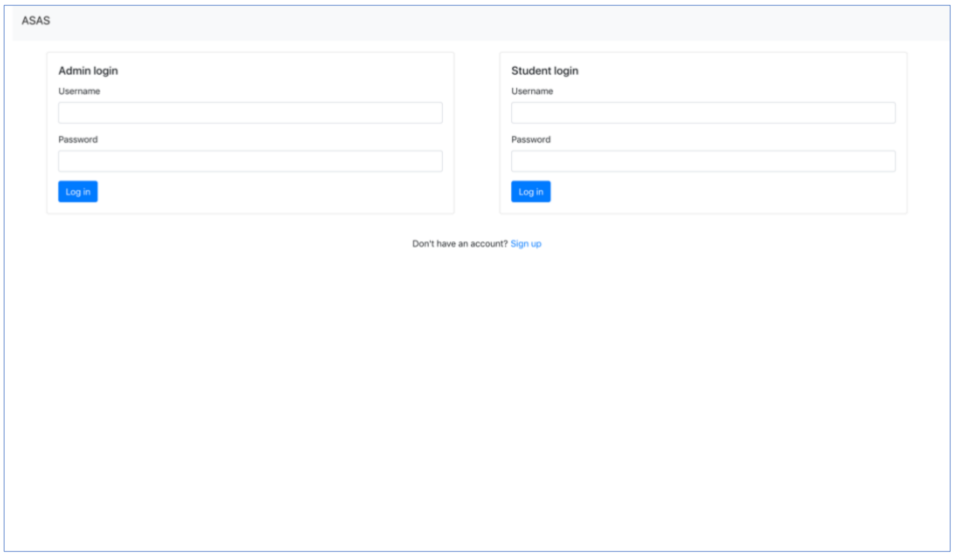
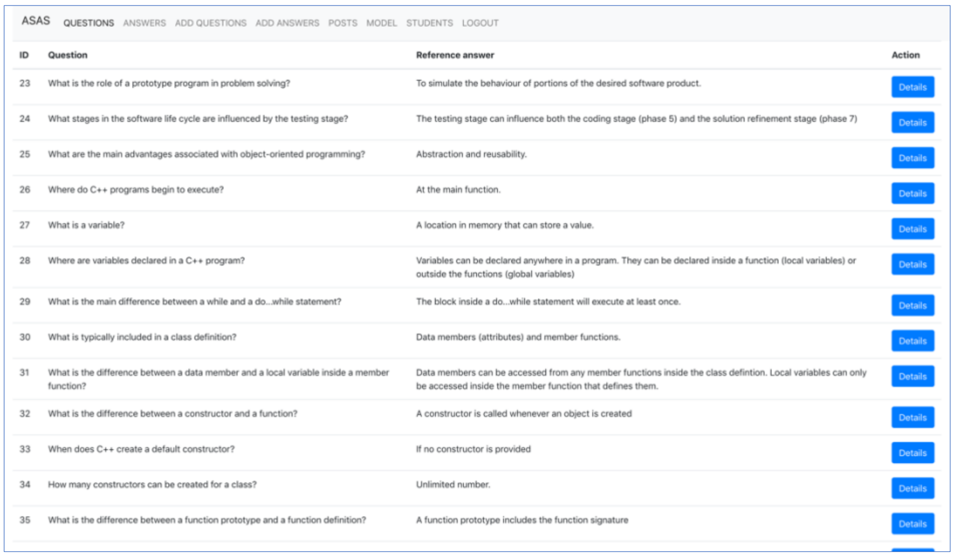


*Figure 13 Log in page*

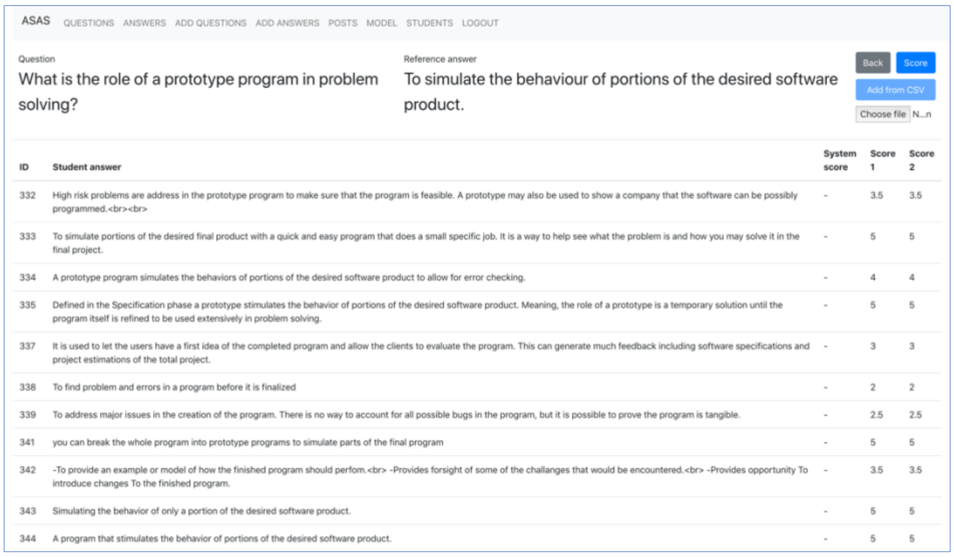
### 5.2.2 Admin

### 5.2.2.1 Questions



| ID | Question | Reference answer | Action |
|---|---|---|---|
| 23 | What is the role of a prototype program in problem solving? | To simulate the behaviour of portions of the desired software product. | Details |
| 24 | What stages in the software life cycle are influenced by the testing stage? | The testing stage can influence both the coding stage (phase 5) and the solution refinement stage (phase 7) | Details |
| 25 | What are the main advantages associated with object-oriented programming? | Abstraction and reusability. | Details |
| 26 | Where do C++ programs begin to execute? | At the main function. | Details |
| 27 | What is a variable? | A location in memory that can store a value. | Details |
| 28 | Where are variables declared in a C++ program? | Variables can be declared anywhere in a program. They can be declared inside a function (local variables) or outside the functions (global variables) | Details |
| 29 | What is the main difference between a while and a do...while statement? | The block inside a do...while statement will execute at least once. | Details |
| 30 | What is typically included in a class definition? | Data members (attributes) and member functions. | Details |
| 31 | What is the difference between a data member and a local variable inside a member function? | Data members can be accessed from any member functions inside the class defintion. Local variables can only be accessed inside the member function that defines them. | Details |
| 32 | What is the difference between a constructor and a function? | A constructor is called whenever an object is created | Details |
| 33 | When does C++ create a default constructor? | If no constructor is provided | Details |
| 34 | How many constructors can be created for a class? | Unlimited number. | Details |
| 35 | What is the difference between a function prototype and a function definition? | A function prototype includes the function signature | Details |

*Figure 14 Admin questions list*

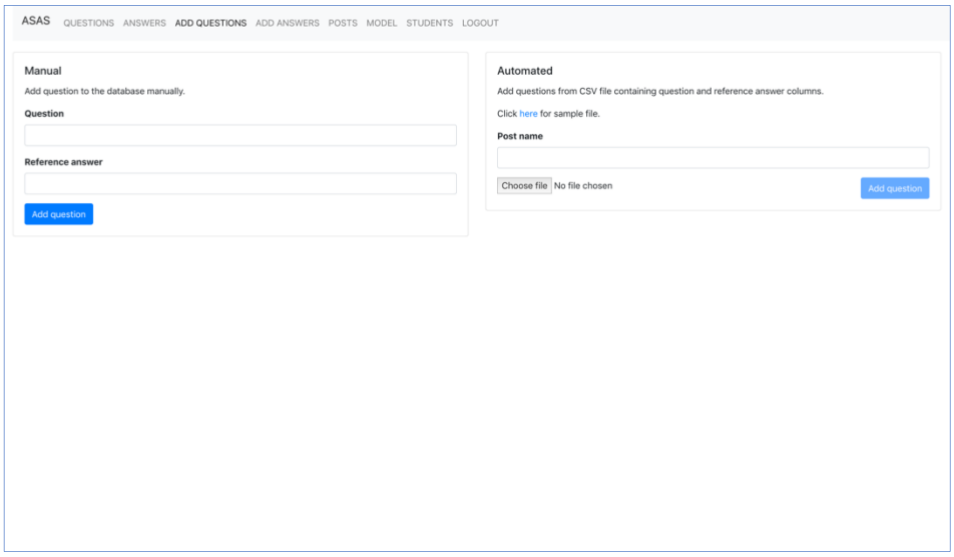


Question
What is the role of a prototype program in problem solving?

Reference answer
To simulate the behaviour of portions of the desired software product.

| ID | Student answer | System score | Score 1 | Score 2 |
|---|---|---|---|---|
| 332 | High risk problems are address in the prototype program to make sure that the program is feasible. A prototype may also be used to show a company that the software can be possibly programmed.<br><br> | - | 3.5 | 3.5 |
| 333 | To simulate portions of the desired final product with a quick and easy program that does a small specific job. It is a way to help see what the problem is and how you may solve it in the final project. | - | 5 | 5 |
| 334 | A prototype program simulates the behaviors of portions of the desired software product to allow for error checking. | - | 4 | 4 |
| 335 | Defined in the Specification phase a prototype stimulates the behavior of portions of the desired software product. Meaning, the role of a prototype is a temporary solution until the program itself is refined to be used extensively in problem solving. | - | 5 | 5 |
| 337 | It is used to let the users have a first idea of the completed program and allow the clients to evaluate the program. This can generate much feedback including software specifications and project estimations of the total project. | - | 3 | 3 |
| 338 | To find problem and errors in a program before it is finalized | - | 2 | 2 |
| 339 | To address major issues in the creation of the program. There is no way to account for all possible bugs in the program, but it is possible to prove the program is tangible. | - | 2.5 | 2.5 |
| 341 | you can break the whole program into prototype programs to simulate parts of the final program | - | 5 | 5 |
| 342 | -To provide an example or model of how the finished program should perfom.<br> -Provides forsight of some of the challanges that would be encountered.<br> -Provides opportunity To introduce changes To the finished program. | - | 3.5 | 3.5 |
| 343 | Simulating the behavior of only a portion of the desired software product. | - | 5 | 5 |
| 344 | A program that stimulates the behavior of portions of the desired software product. | - | 5 | 5 |

*Figure 15 Admin question details*



Manual

Add question to the database manually.

Question

Reference answer

Automated

Add questions from CSV file containing question and reference answer columns.

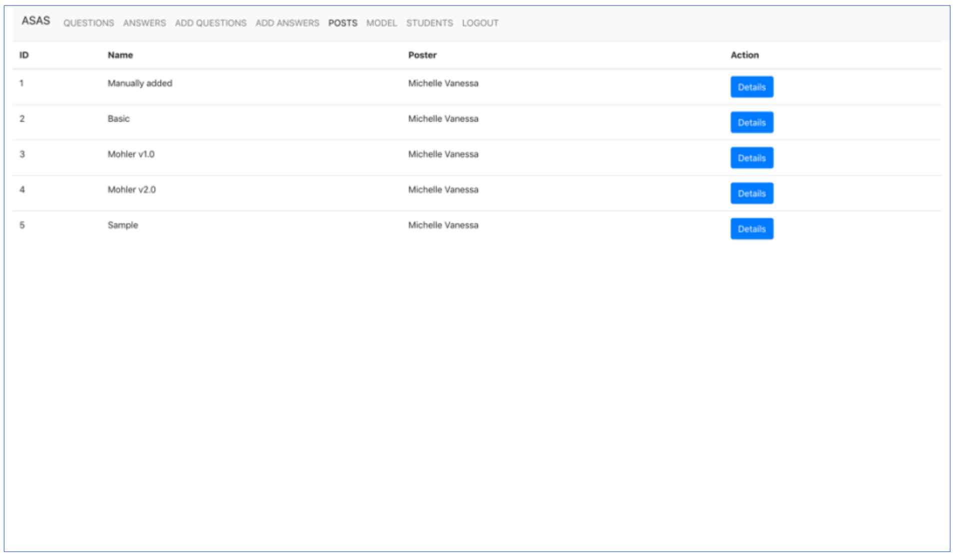
Click here for sample file.

Post name

No file chosen

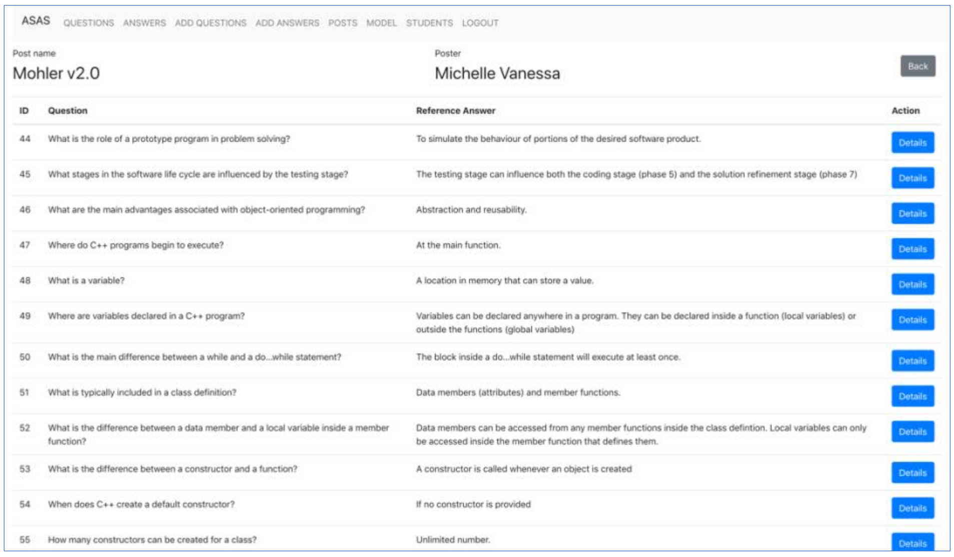
*Figure 16 Add questions*

### 5.2.2.2 Posts



| ID | Name | Poster | Action |
|---|---|---|---|
| 1 | Manually added | Michelle Vanessa | Details |
| 2 | Basic | Michelle Vanessa | Details |
| 3 | Mohler v1.0 | Michelle Vanessa | Details |
| 4 | Mohler v2.0 | Michelle Vanessa | Details |
| 5 | Sample | Michelle Vanessa | Details |

*Figure 17 Admin posts list*



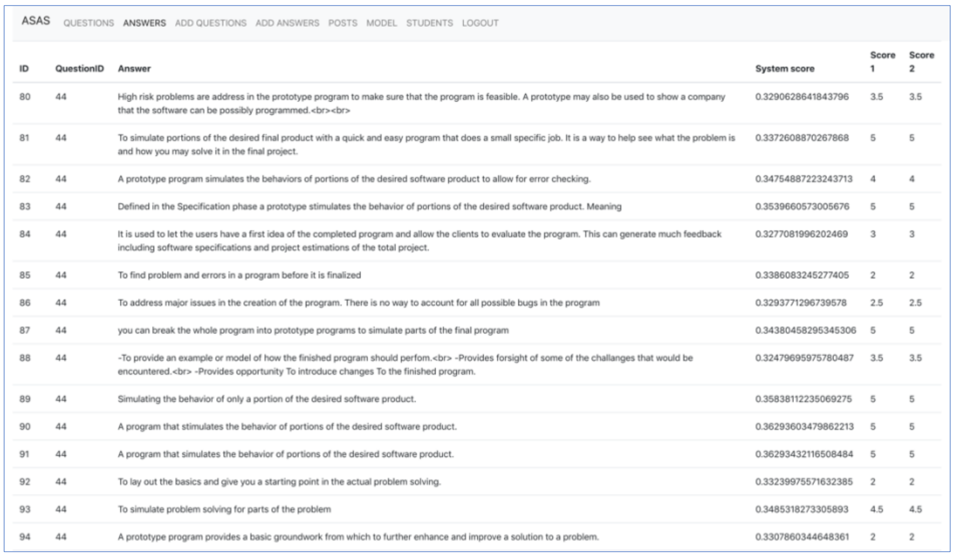
Post name
Mohler v2.0

Poster
Michelle Vanessa

| ID | Question | Reference Answer | Action |
|---|---|---|---|
| 44 | What is the role of a prototype program in problem solving? | To simulate the behaviour of portions of the desired software product. | Details |
| 45 | What stages in the software life cycle are influenced by the testing stage? | The testing stage can influence both the coding stage (phase 5) and the solution refinement stage (phase 7) | Details |
| 46 | What are the main advantages associated with object-oriented programming? | Abstraction and reusability. | Details |
| 47 | Where do C++ programs begin to execute? | At the main function. | Details |
| 48 | What is a variable? | A location in memory that can store a value. | Details |
| 49 | Where are variables declared in a C++ program? | Variables can be declared anywhere in a program. They can be declared inside a function (local variables) or outside the functions (global variables) | Details |
| 50 | What is the main difference between a while and a do...while statement? | The block inside a do...while statement will execute at least once. | Details |
| 51 | What is typically included in a class definition? | Data members (attributes) and member functions. | Details |
| 52 | What is the difference between a data member and a local variable inside a member function? | Data members can be accessed from any member functions inside the class defintion. Local variables can only be accessed inside the member function that defines them. | Details |
| 53 | What is the difference between a constructor and a function? | A constructor is called whenever an object is created | Details |
| 54 | When does C++ create a default constructor? | If no constructor is provided | Details |
| 55 | How many constructors can be created for a class? | Unlimited number. | Details |

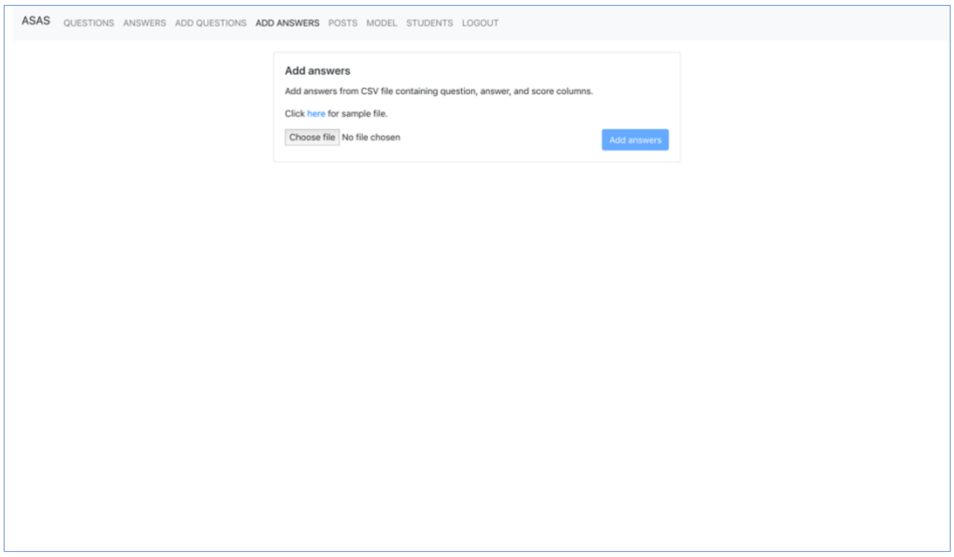
*Figure 18 Admin post details*

### 5.2.2.3 Answers



| ID | QuestionID | Answer | System score | Score 1 | Score 2 |
|---|---|---|---|---|---|
| 80 | 44 | High risk problems are address in the prototype program to make sure that the program is feasible. A prototype may also be used to show a company that the software can be possibly programmed.<br><br> | 0.3290628641843796 | 3.5 | 3.5 |
| 81 | 44 | To simulate portions of the desired final product with a quick and easy program that does a small specific job. It is a way to help see what the problem is and how you may solve it in the final project. | 0.3372608870267868 | 5 | 5 |
| 82 | 44 | A prototype program simulates the behaviors of portions of the desired software product to allow for error checking. | 0.34754887223243713 | 4 | 4 |
| 83 | 44 | Defined in the Specification phase a prototype stimulates the behavior of portions of the desired software product. Meaning | 0.3539660573005676 | 5 | 5 |
| 84 | 44 | It is used to let the users have a first idea of the completed program and allow the clients to evaluate the program. This can generate much feedback including software specifications and project estimations of the total project. | 0.3277081996202469 | 3 | 3 |
| 85 | 44 | To find problem and errors in a program before it is finalized | 0.3386083245277405 | 2 | 2 |
| 86 | 44 | To address major issues in the creation of the program. There is no way to account for all possible bugs in the program | 0.3293771296739578 | 2.5 | 2.5 |
| 87 | 44 | you can break the whole program into prototype programs to simulate parts of the final program | 0.34380458295345306 | 5 | 5 |
| 88 | 44 | -To provide an example or model of how the finished program should perfom.<br> -Provides forsight of some of the challanges that would be encountered.<br> -Provides opportunity To introduce changes To the finished program. | 0.32479695975780487 | 3.5 | 3.5 |
| 89 | 44 | Simulating the behavior of only a portion of the desired software product. | 0.35838112235069275 | 5 | 5 |
| 90 | 44 | A program that stimulates the behavior of portions of the desired software product. | 0.36293603479862213 | 5 | 5 |
| 91 | 44 | A program that simulates the behavior of portions of the desired software product. | 0.36293432116508484 | 5 | 5 |
| 92 | 44 | To lay out the basics and give you a starting point in the actual problem solving. | 0.33239975571632385 | 2 | 2 |
| 93 | 44 | To simulate problem solving for parts of the problem | 0.3485318273305893 | 4.5 | 4.5 |
| 94 | 44 | A prototype program provides a basic groundwork from which to further enhance and improve a solution to a problem. | 0.3307860344648361 | 2 | 2 |

*Figure 19 Admin answers list*


ASAS
QUESTIONS ANSWERS ADD QUESTIONS ADD ANSWERS POSTS MODEL STUDENTS LOGOUT

Add answers

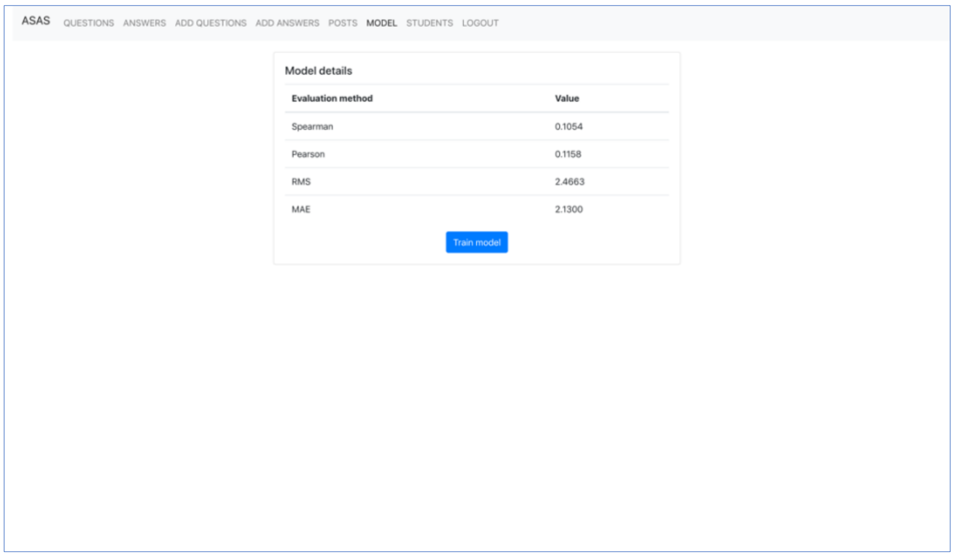
Add answers from CSV file containing question, answer, and score columns.

Click here for sample file.

Choose file No file chosen
Add answers

*Figure 20 Add answers*

#### 5.2.2.4 Model

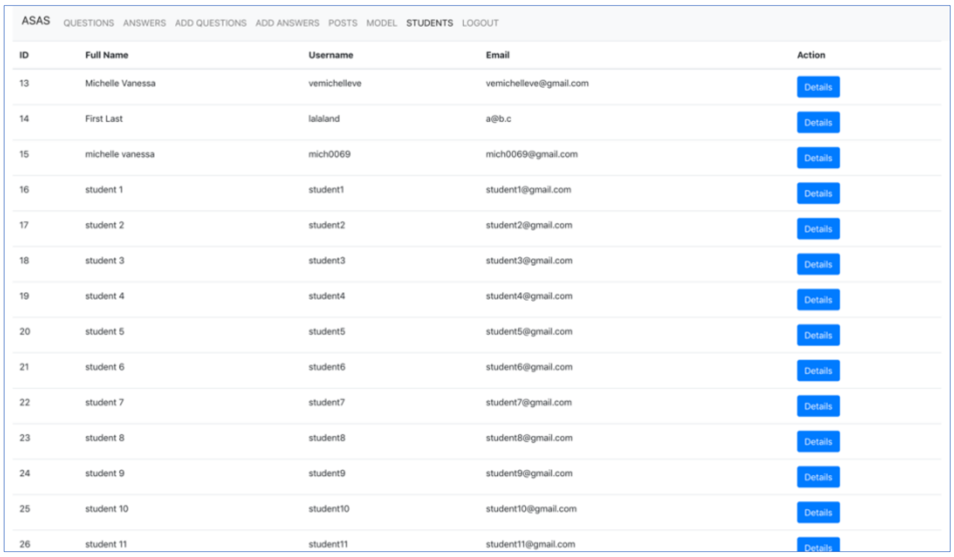

ASAS
QUESTIONS ANSWERS ADD QUESTIONS ADD ANSWERS POSTS MODEL STUDENTS LOGOUT

Model details

| Evaluation method | Value |
|---|---|
| Spearman | 0.1054 |
| Pearson | 0.1158 |
| RMS | 2.4663 |
| MAE | 2.1300 |

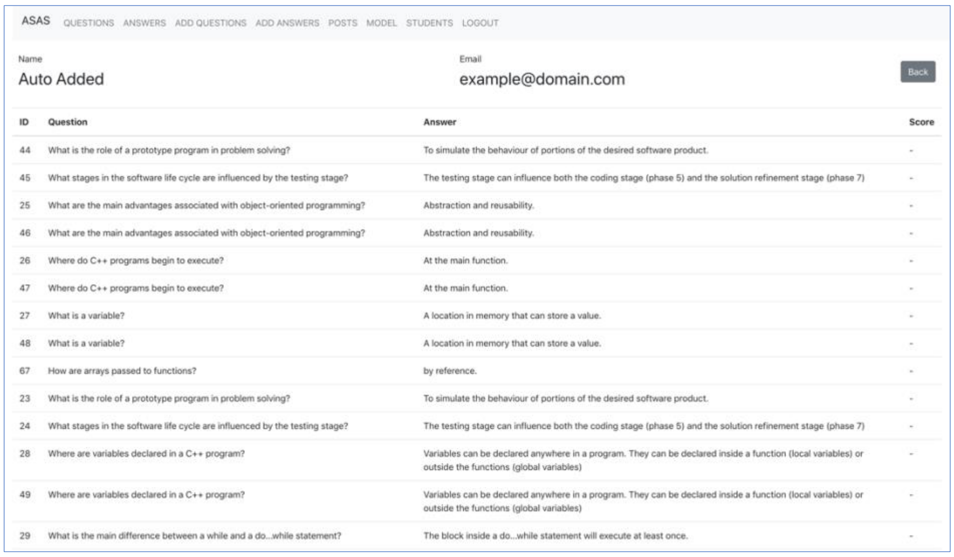
*Figure 21 Model details*

#### *5.2.2.5 Students*



| ID | Full Name | Username | Email | Action |
|---|---|---|---|---|
| 13 | Michelle Vanessa | vemichelleve | vemichelleve@gmail.com | Details |
| 14 | First Last | lalaland | a@b.c | Details |
| 15 | michelle vanessa | mich0069 | mich0069@gmail.com | Details |
| 16 | student 1 | student1 | student1@gmail.com | Details |
| 17 | student 2 | student2 | student2@gmail.com | Details |
| 18 | student 3 | student3 | student3@gmail.com | Details |
| 19 | student 4 | student4 | student4@gmail.com | Details |
| 20 | student 5 | student5 | student5@gmail.com | Details |
| 21 | student 6 | student6 | student6@gmail.com | Details |
| 22 | student 7 | student7 | student7@gmail.com | Details |
| 23 | student 8 | student8 | student8@gmail.com | Details |
| 24 | student 9 | student9 | student9@gmail.com | Details |
| 25 | student 10 | student10 | student10@gmail.com | Details |
| 26 | student 11 | student11 | student11@gmail.com | Details |

*Figure 22 Students list*



Name
Auto Added

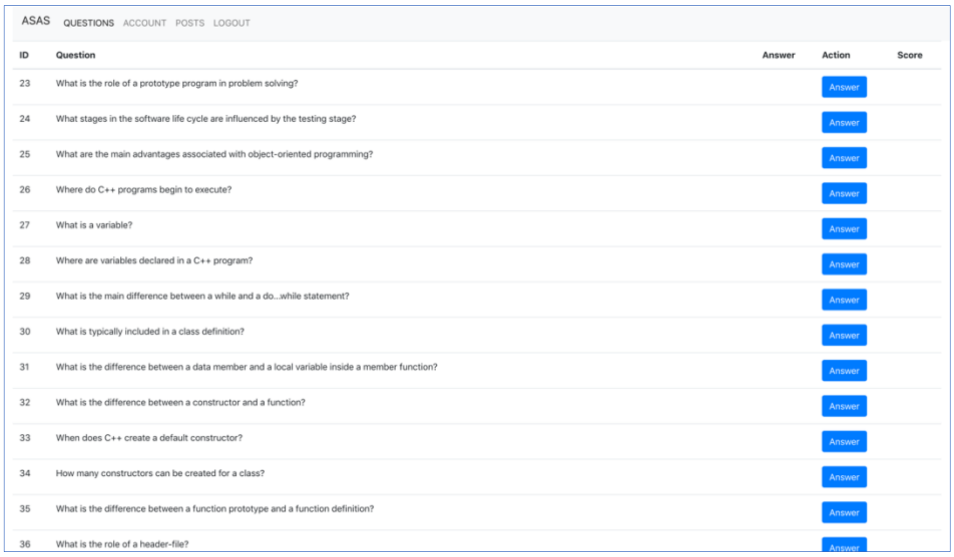
Email
example@domain.com

| ID | Question | Answer | Score |
|---|---|---|---|
| 44 | What is the role of a prototype program in problem solving? | To simulate the behaviour of portions of the desired software product. | - |
| 45 | What stages in the software life cycle are influenced by the testing stage? | The testing stage can influence both the coding stage (phase 5) and the solution refinement stage (phase 7) | - |
| 25 | What are the main advantages associated with object-oriented programming? | Abstraction and reusability. | - |
| 46 | What are the main advantages associated with object-oriented programming? | Abstraction and reusability. | - |
| 26 | Where do C++ programs begin to execute? | At the main function. | - |
| 47 | Where do C++ programs begin to execute? | At the main function. | - |
| 27 | What is a variable? | A location in memory that can store a value. | - |
| 48 | What is a variable? | A location in memory that can store a value. | - |
| 67 | How are arrays passed to functions? | by reference. | - |
| 23 | What is the role of a prototype program in problem solving? | To simulate the behaviour of portions of the desired software product. | - |
| 24 | What stages in the software life cycle are influenced by the testing stage? | The testing stage can influence both the coding stage (phase 5) and the solution refinement stage (phase 7) | - |
| 28 | Where are variables declared in a C++ program? | Variables can be declared anywhere in a program. They can be declared inside a function (local variables) or outside the functions (global variables) | - |
| 49 | Where are variables declared in a C++ program? | Variables can be declared anywhere in a program. They can be declared inside a function (local variables) or outside the functions (global variables) | - |
| 29 | What is the main difference between a while and a do...while statement? | The block inside a do...while statement will execute at least once. | - |

*Figure 23 Student details*

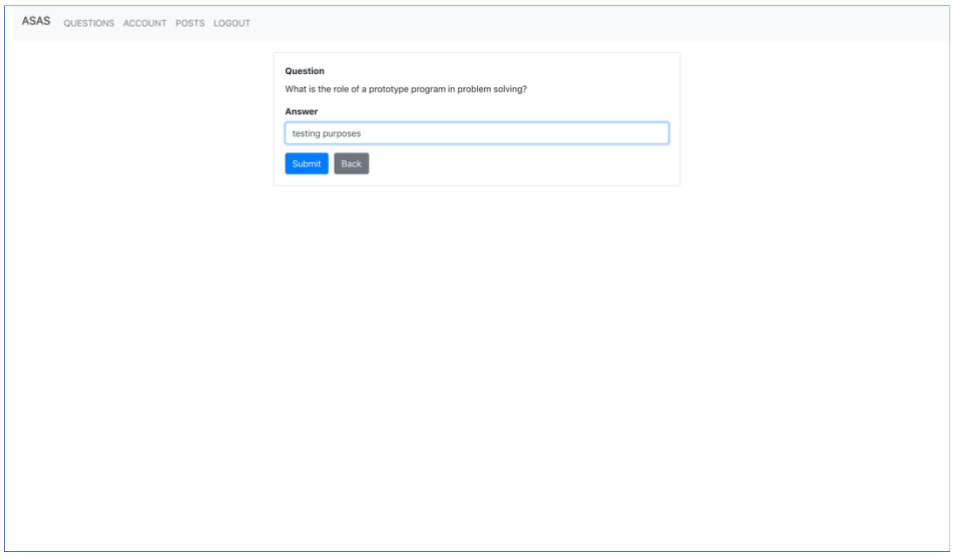
### 5.2.3 Student

### 5.2.3.1 Questions



| ID | Question | Answer | Action | Score |
|---|---|---|---|---|
| 23 | What is the role of a prototype program in problem solving? | | Answer | |
| 24 | What stages in the software life cycle are influenced by the testing stage? | | Answer | |
| 25 | What are the main advantages associated with object-oriented programming? | | Answer | |
| 26 | Where do C++ programs begin to execute? | | Answer | |
| 27 | What is a variable? | | Answer | |
| 28 | Where are variables declared in a C++ program? | | Answer | |
| 29 | What is the main difference between a while and a do...while statement? | | Answer | |
| 30 | What is typically included in a class definition? | | Answer | |
| 31 | What is the difference between a data member and a local variable inside a member function? | | Answer | |
| 32 | What is the difference between a constructor and a function? | | Answer | |
| 33 | When does C++ create a default constructor? | | Answer | |
| 34 | How many constructors can be created for a class? | | Answer | |
| 35 | What is the difference between a function prototype and a function definition? | | Answer | |
| 36 | What is the role of a header-file? | | Answer | |

*Figure 24 Student questions list*



**Question**

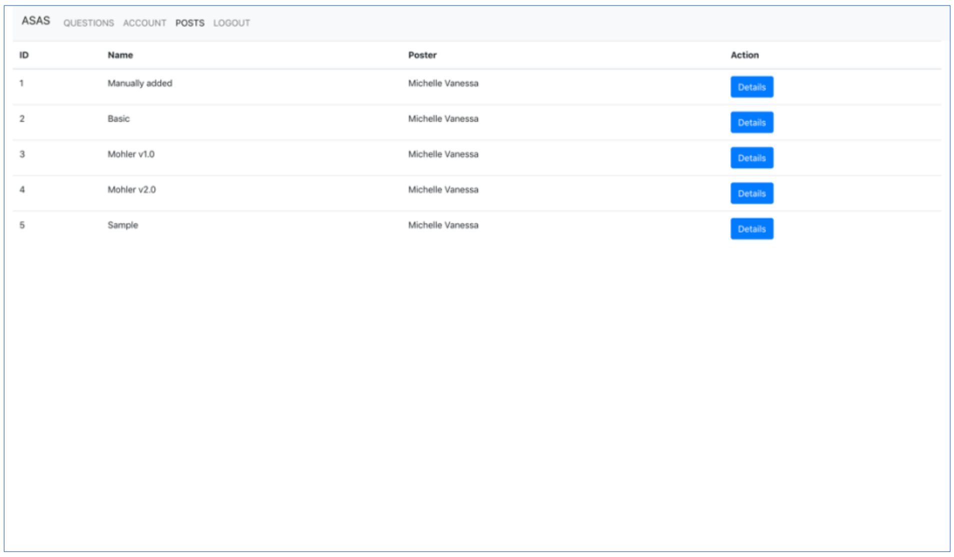
What is the role of a prototype program in problem solving?

**Answer**

testing purposes

*Figure 25 Student answer question page*

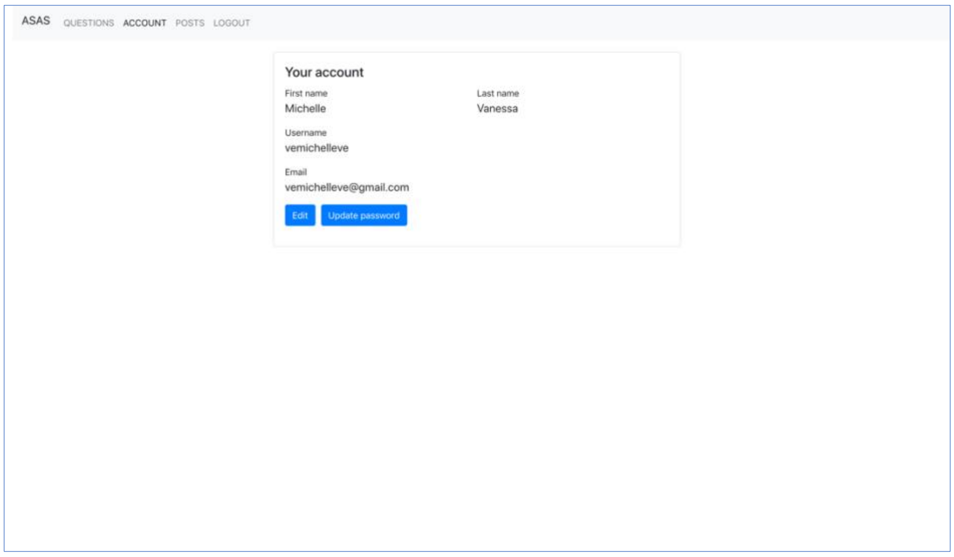
### 5.2.3.2 Posts



| ID | Name | Poster | Action |
|---|---|---|---|
| 1 | Manually added | Michelle Vanessa | Details |
| 2 | Basic | Michelle Vanessa | Details |
| 3 | Mohler v1.0 | Michelle Vanessa | Details |
| 4 | Mohler v2.0 | Michelle Vanessa | Details |
| 5 | Sample | Michelle Vanessa | Details |

*Figure 26 Student posts list*

### 5.2.3.3 Account



Your account

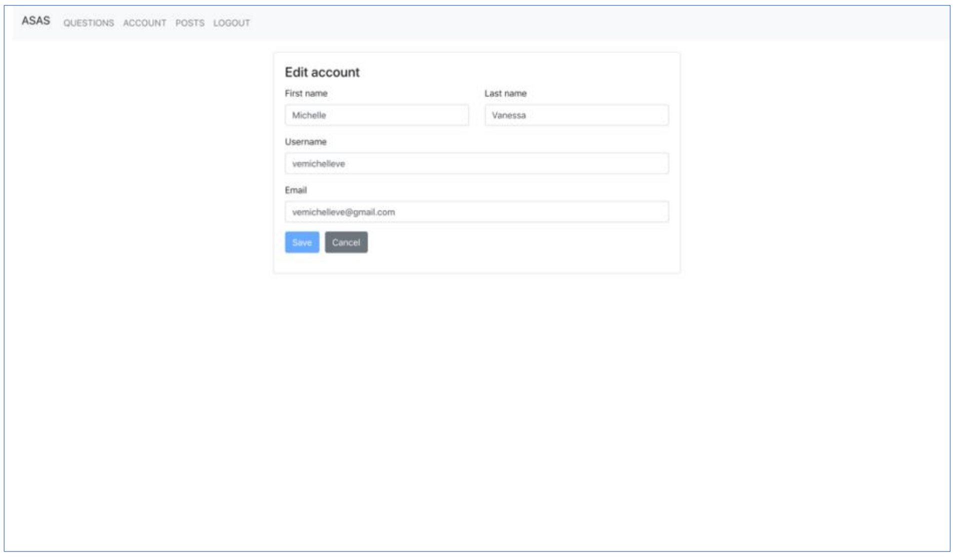
First name
Michelle

Last name
Vanessa

Username
vemichelleve

Email
vemichelleve@gmail.com

*Figure 27 Student account details*



Edit account

First name: Michelle

Last name: Vanessa

Username: vemichelleve

Email: vemichelleve@gmail.com

*Figure 28 Student edit account*

# Chapter 6: Conclusion

# Reference

[1] M. Mohler, R. Bunescu and R. Mihalcea, “Learning to Grade Short Answer Questions using Semantic Similarity Measures and Dependency Graph Alignments,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, Protland, Oregon, 2011.

[2] M. Zhang, “Contrasting Automated and Human Scoring of Essays,” in *R&D Connection*, vol. 21, Princeton, NJ: Research & Development Educational Testing Service, 2013, pp. 1-11.

[3] C. Leacock and M. Chodorow, “C-rater: Automated Scoring of Short Answer Questions,” *Computer and Humanities*, vol. 37, pp. 389-405, 2003.

[4] M. Mohler and R. Mihalcea, “Text-to-text Semantic Similarity for Automatic Short Answer Grading,” in *Proceedings of the 12th Conference of the European Chapter of the ACL*, Athens, Greece, 2009.

[5] F. S. Pribadi, T. B. Adji, A. E. Permanasari, A. Mulwinda and A. B. Utomo, “Automatic Short Answer Scoring Using Words Overlapping Methods,” in *Engineering International Conference*, 2016.

[6] S. Hassan, A. A. Fahmy and M. El-Ramly, “Automatic Short Answer Scoring based on Paragraph Embeddings,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 10, pp. 397-402, 2018.

[7] M. A. Sultan, C. Salazar and T. Sumner, “Fast and Easy Short Answer Grading with High Accuracy,” in *Proceedings of NAACL-HLT*, San Diego, California, 2016.

[8] C. Olah, “Understanding LSTM Networks,” 27 August 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 10 March 2020].

[9] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[10] S. Santurkar, D. Tsipras, A. Ilyas and A. Madry, “How Does Batch Normalization Help Optimization?,” 2018.

[11] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” 2015.

[12] G. E. Dahl, T. N. Sainath and G. E. Hinton, “Improving Deep Neural Networks for LVCSR Using Rectified Linear Units and Dropout,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing,* 2013.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, “Improving Neural Networks by Preventing Co-adaptation of Feature Detectors,” 2012.

[14] Y. Goldberg and O. Levy, “word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method,” 2014.

[15] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” *Advances in Neural Information Processing Systems 26,* 2013.

[16] A. Casari and A. Zheng, Feature Engineering for Machine Learning, Sebastopol: O’Reilly Media, Inc., 2018.

[17] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[18] Z. Huang, W. Xu and K. Yu, “Bidirectional LSTM-CRF Models for Sequence Tagging,” 2015.

[19] J. Bjorck, C. Gomes, B. Selamn and K. Q. Weinberger, “Understanding Batch Normalization,” 2018.

[20] D. Ho, E. Liang and R. Liaw, “1000x Faster Data Augmentation,” Berkeley Artificial IntelligentceResearch, 7 June 2019. [Online]. Available: https://bair.berkeley.edu/blog/2019/06/07/data_aug/. [Accessed 11 March 2020].

[21] J. Perla, “Notes on AdaGrad,” 2014.

[22] “Regularization for Simplicity: $L_2$ Regularization,” Machine Learning Crash Course, 10 February 2020. [Online]. Available: https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/l2-regularization. [Accessed 12 March 2020].

[23] “Django,” [Online]. Available: https://www.djangoproject.com/. [Accessed 14 March 2020].

[24] “Django Documentation,” [Online]. Available: https://docs.djangoproject.com/en/3.0/. [Accessed 14 March 2020].

[25] “Django REST Framework,” [Online]. Available: https://www.django-rest-framework.org/. [Accessed 14 March 2020].