Explaining and Interpreting Model Predictions for Short Answer Grading*

Ravikiran Bhat, Deebul Nair[1] and Paul G. Plöger[2]

***Abstract*— This work focuses on associating a reasoning for automated scores for short answers type questions as a means of rationalizing the predicted score to the human graders. Observations resulting from the pre-investigations conducted on two state-of-the-art model-agnostic interpretation techniques, namely, LIME and SHAP serves as the inspiration in the development of our solution. In our work, by relying on a simple BOW model supported by active learning, we develop an algorithm and a grading assisting tool that supports human graders with automated grading together with highlighted features acting as supporting explanations to enable the human graders to interpret the cause of the suggested grades.**

I. INTRODUCTION

Assessment is a task that is an integral component of learning process in educational institutions and involves a human grader evaluating the results of anexamination process[8][9]. Handling situations where there are a limited numbers of graders, or a large number of students, and also to eliminate monotonic and bias errors resulting from manual assessments made it necessary to include a form of automatic scoring system to replace manual assessments that guaranteed fairness and saved teachers time[10]. Automated scoring systems for long answer essays and short answer type questions that require answers to be constructed in natural language form mostly work by comparing students answer to a model answer. Being efficient and cost-effective solutions, these systems produce good results in terms of consistency by reducing the imperfection resulting from fatigue, bias, or ordering effects of the human raters[11]. However since open-style questions can elicit responses from students in more than one form due to the students not having the comfort of being able to recall the exact words or phrases matching the model answers, the absence of some form of diagnostic feedback to the human raters by the automatic short answer grading (ASAG) systems as a way of justifying the given grades[4] is a drawback in current ASAG systems.

This work aims at addressing the problem of determining the applicability of model interpretation as a solution towards satisfying the requirement of generating convincing justifications for autograded answers. We also include a form of active learning throught the grading process to leverage the abilities of both the human grader and the machine, so that the process of grading of short answers is turned into a form of *automated assistance* by the system to the human graders to help speed up the grading process rather than have a fully automated scoring system. The initial part of this research involves conducting preinvestigations on two state of the art model interpretation techniques and their applicability towards short answer grading. We use the insights gained from the investigations to propose our approach that uses a simple bag-of-words model in conjunction with active learning to provide automated grades with highlighted feedback to the human grader. A web GUI is developed to integrate our proposed solution, and a final survey of the tool with the teaching staff is performed to analyze the usefulness of the developed grading assistance tool and identify the requirements for out future improvements.

[1]Deebul Nair is with Faculty of Computer Science, University of Applied Sciences Bonn-Rhein-Sieg, Sankt Augustin, Germany. deebul-sivarajan.nair@h-brs.de

[2]Paul G. Plöger is the Vicedean of the Department of Computer Science, Autonomous Systems, University of Applied Sciences Bonn-Rhein-Sieg, Sankt Augustin, Germany. paul.ploeger@h-brs.de

II. RELATED WORK

The methodology presented in [1] is the closest related work to our research and is also one of the most promising works in the field of model interpretation and explanations. The *LIME* (Local interpretable model-agnostic explanations) method presented in this paper approximates a black-box model locally in the neighborhood of a specific prediction, i.e, it fits local, interpretable models that can explain predictions for a specific instance of *any* classifier. The novelty behind LIME is that the model agnostic explanations of LIME are produced in the form of color coded features in the text that are easily understood even by human users with no underlying knowledge of the working of the model used to classify the predictions. These explanations are useful in many scenarios that require promoting trust in a prediction and is helpful in determining whether a classifier should be trusted or not. Another model-agnostic algorithm introduced by the same researchers of *LIME* involves the use of "anhors" where a perturbation strategy similar to *LIME* is used to generate local explantions for black box model predictions, but instead of using surrogate model as in *LIME*, the explanations are expressed in the form of easily understood *IF-THEN* rules called *anchors*[3][4].

Similarly belonging to the class of model-agnnostic model interpretation methods is the SHAP (SHapley Additive exPlanations) framework presented by Lundberg et. al in [2] that leverages the concept of Shapley values explanations from cooperative game theory. The SHAP method presented in the paper assigns each feature an importance value and use them to explain predictions. Since method of computing the SHAP values involves the exhaustive approach of computing the "average marginal contribution of a feature value over all possible coalitions"[3], it theoretically guarantees the

properties of accuracy and consistency.
Other model-agnostic methods such as counterfactual explanations are relatively more constrained in that they belong to a class of example-based explantions and they mainly work by selecting instances of the dataset to explain a model[3]. Counterfactual explanations are useful in explaining predictions of individual instances in interpretable machine learning by expressing causal situations in the form: "If X had not occurred, Y would not have occurred". Martens et. al in [5] extends this theory to document classification where they try to explain why a document was or was not classified as a particular class. More specifically, a document D's classification is explained as a set of words E such that removing all words in E leads to the classifier producing a different classification. In addition, this explanation is minimal in the sense that removing a subset of E from the document will not change the classification.
Layer wise relevance propagation was a notable approach introduced to interpret predictions of deep neural network architectures[6] where the basic idea invloved using hte back propagazion procedure to redistribute the output prediction score causing the classification, back to input space. A a scalar relevance value assigned for each input variable by summing over the the input dimension of weighted relevance values for each neuron indicates how much the corresponding input contributes to the overall classification decision. The tree interpreter algorithm, a concept developed by Saabas[7] is another useful algorithm, although it is specialized to interpret Random Forest predictions. The main idea here is to determine contributions of the features used for a particular prediction by tracing the decision paths of a forest and each feature's contributions is elucidated by calculating how the training set mean changes on following a prediction path.

III. PRE-INVESTIGATIONS

This section describes a series of investigative experiments which are responsible for determining the main areas of focus in our solution design. Two of the existing model interpretation techniques from literature, namely LIME and SHAP were the main focus of the experiments. All experiments are carried out via a supervised learning process with 65% of the data used as the training set and the rest for testing. We use a Bag of Words (BOW) approach as the basis for extracting ngrams and use them as predictors for the scores and Naive Bayes classifier was used as the learning model to classify the discrete features. The the grading is treated as a binary classification task with each answer graded as correct or incorrect. The observations from the experiments were used to narrow down the shortcomings of the two model interpretation techniques that we attempt to address in our future solution design. The experiments in this section therefore serve as a benchmark that heavily influences the direction of our approach discussed in the next section.

A. Datasets

All experiments in this section were conducted on datasets from two sources:

1) **Mohler's Dataset**
   The dataset consists of a set of questions taken from assignments given out for the Data Structure course at the University of North Texas[8]. The answers were collected from a class of 31 undergraduate students via an online learning assignment. The dataset was created by working on student answers for 80 different questions and a total of 2273 answers. The answers were scored independently by two human graders on scale ranging from 0 (completely incorrect) to 5 (perfect answer). In our pre investigations, student scores above 3 were labeled as 'Correct' and those graded 3 or lower were labeled as 'Incorrect'.
2) **Neural Network Dataset**
   This dataset consists of 17 questions administered as part of the Neural Networks course examination at the University of applied sciences Bonn-Rhein-Sieg. The answers were collected from a class of 39 students from an online examination conducted using Jupyter Notebooks. The dataset was created by using the student answers for the first 17 questions which required mandatory answers. The answers were scored on an integer scale as either 0(completely incorrect), 1 (partially correct) or 2 (perfect answer). In our pre investigations, student scores below 2 points were labelled as 'Incorrect' and those with 2 points were labelled as 'Correct'.

A total of 14 questions selected from both datasets acted as case studies in our experiments. The questions selected included those which required answers in a few words (i.e not exceeding 4-5 words), in one or two sentences, and those where answers were longer than two to three sentences. A sub goal of the investigations in this section was to narrow down the attributes of the questions for which the applied interpretability techniques gives the best results matching human expectations.
A separate model is learned after extracting features from the answers to each question. Before extracting features and learning a model, the answers from the two datasets are subject to preprocessing using NLP techniques where the model answers and student answers were invloving tokenizing the words and convertion into lowercase form, stopword removal and lemmatization to obtain the base / dictionary form of a word.

B. Benchmarking Experiments

Two experiments are conducted to check the differences in the degree of understanding gained from the explanations given by LIME and SHAP.

1) *Experiment 1*:
   In this experiment, LIME is applied on the unseen test instances to get an explanation that is locally faithful to the classifier. This experiment is divided into 2 parts. In the first part, we experiment with two feature extractors, namely 'CountVectorizer' which gives a a sparse matrix representation of the token counts from the student answers, and the TfidfVectorizer that gives

a normailzed score for each of the word occurance count. The goal of the first part of the experiment is determine the best feature extractor by observing the influence on the final LIME explanation. In the second part of the experiment, we focus on the explanation obtained for every test instance for every case study to determine the degree of understanding gained for different answers, and also narrow down the limitations in the explanations that needs to be addressed.

2) *Experiment 2*:
In this experiment, we apply Shapley value explanations as the model-agnostic interpretation technique for our predicted scores. The SHAP (SHapley Additive exPlanations) library introduced by Lundberg et al.[2] is used to visualize a global overview of the impact of the top k features on the model output. The goal of this experiment is to analyze the global interpretation obtained from SHAP and contrast it with the locally faithful explanations generated by LIME.

C. Benchmarking Experiments' Results and Analysis

Experiment 1:
Results of experiment 1 showed that, contrary to expectations, performance in terms of the final accuracy as well as the explanations from LIME were identical for both CountVectorizer and TfidfVectorizer. Observing the predictions and explanations on unseen data, it was seen that while very few instances showed some minor differences in the highlighting of unigrams, these differences did not have any noticeable impact on the interpretation gained by the end user. Some other general observations on the LIME explanations summarized below:

- The shorter the answer length, the closer the highlighted unigrams was to human expectations.
- Lower model accuracies consequently ended up affecting not only the predicted scores, but the resulting LIME highlighted explanations also tended to being more unacceptable by humans.
- Larger answers and higher linguistic diversity in the student responses resulted in explanations that were inconceivable due to LIME's limitations in highlighting only unigrams in the text.

Experiment 2:
We rely on the SHAP's summary plot to get a display of the top 20 globally important features affecting the model output for every answer in the test set to a given question. An example summary plot for the class "Correct" showing the SHAP value for the most important features identified from among the test samples to the question: *"What is the role of prototype program in problem solving?"* is shown in Figure 1 that has the following model answer in the dataset: *"To simulate the behaviour of portions of desired software product"*.

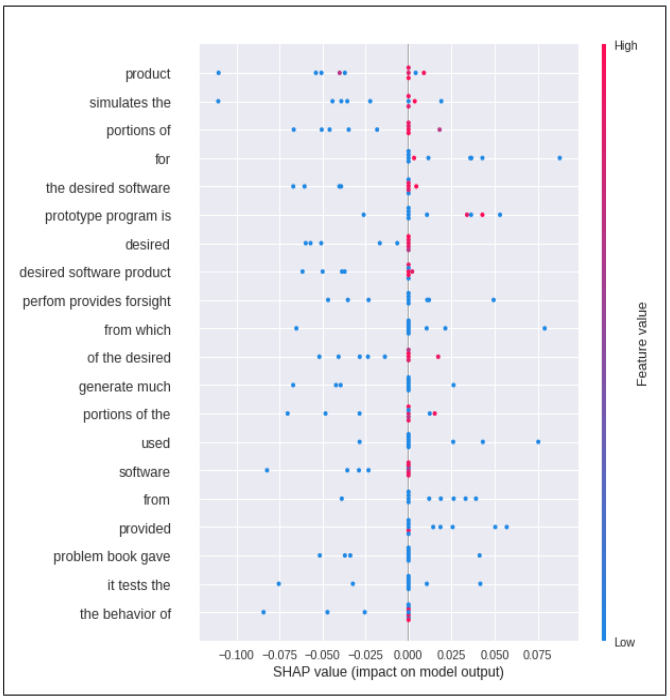

Fig. 1. An example SHAP summary plot of top 20 features for the class "Correct".

Each dot in the plot represents one instance from the test set. The plot reveals that low values (i.e, the samples highlighted in blue) of features such as *"simulates the"*, *"portions of"*, *"desired"* , *"software"*, etc for samples (low here indicating the absence of said feature) will have negative influence on model output. This indicates that these features are important for a student answer to be classified as *"Correct"*. Presence of these features in any sample (highlighted in red) will not influence the classification of the answer as *"Correct"* in a negative way. This is an example of a case where a human's intuition of the features that are the most important predictors towards a particular class matches closely with the explanation displayed by the summary plot. After observing the summary plots for all the questions selected as our case studies, we come to the following conclusions on the global explanation from SHAP's summary plots:

- Questions with shorter answers(i.e, where length of answers was limited to maximum of one sentence) gave the best results in terms of the top features identified matching human expectations.
- For questoins with longer answers and cases where there was a great amount of linguistic diversity between student answers, the features identified and/or the impacts shown on the model outputs in the plots were redundant and unconvincing. In such cases the global explanations given by the summary plots played no useful role, and great amount of effort from the user was needed to interpret the explanations.

Global interpretation such as the one described in the third experiment is useful in understanding and getting a sense of the magnitude and direction of influence of each feature on

the prediction. Although SHAPs strength lies in providing a globally consistent and accurate explanations by using an exhaustive approach of considering predictions for all possible coalitions for each feature, compared to LIME which is fast and where explanations are obtained instantaneously, computing the Shapley values for the features takes time in the order of 5-10 minutes. On the other hand, while LIME does not have the accuracy and consistency properties of Shapley values, LIME's output provided a more "human friendly" explanation, and felt a more preferable choice when explaining the cause of predictions to end users.

D. Quantitative Benchmarking

In this section, we attempt to quantitatively evaluate and measure the level of interpretation gained from LIME's local explanations and SHAP's global explanation. However, with no clear consensus or ground truth available to compare different interpretations techniques, we developed a strategy whereby an initial qualitative assessment of the explanations was done followed by a mapping to a value within a fixed range to get an idea of how the two interpretation techniques measure against each other in their explanations for the different cases studies. This evaluation is carried out by the experimenter with concrete procedure being as follows:

- When using LIME to get explanations for a student answer, we assign a score on a 3 point Likert scale(i.e, from 1-3) to indicate the degree to which the explanation is acceptable(1 being the lowest, and 3 being the highest score indicating that a human user completely agrees with the explanation.
- In case of SHAP's global explanations from summary plots, we score the explanation based on how accurately we feel the features and their impacts have been captured in the visualization. In case of SHAP, a 3-point Likert scale range was insufficient to capture the level of interpretability, understanding and satisfaction from the explanation obtained by the user, and thus used a wider 5-point scale was used.

Ordinal data captured using the above method was then mapped to a single value in the range [0,1] for each question acting as the case study which served as our interpretation level for comparsion. The mapping was done as follows:

- In case of LIME, after giving an ordinal score for each instance of student answer for a question, we compute the ratio of the occurrences for each type of ordinal score.
- We rely on SciPy librarys interpolation function to perform the following mappings:
  - Ratios computed for ordinal value **1** is mapped from [0, 1] => [0, 0.33] range.
  - Ratios computed for ordinal value **2** is mapped from [0, 1] => [0.34, 0.66] range.
  - Ratios computed for ordinal value **3** is mapped from [0, 1] => [0.67, 1] range.
- The mapping obtained for the ordinal value with the largest ratio in the data for a given question is taken to be the Interpretation Level (IL) for that In case we have a situation where more than one type of ordinal score has the same maximum ratio of occurrences (for e.g ordinal scores 1 and 2 both having a ratio 6 out of a total of 15 samples), in such a case we simply take the average of those ordinal scores, and the average value that lies between 1 and 3 is mapped to [0,1] range directly.
- In case of SHAP, the ordinal data was directly mapped to the [0, 1] range which corresponded to the Interpretation Level (IL) for SHAP for that question.

The Figure 2 shows the comparison of the interpretation levels for every question used as case study from the Mohler's dataset.

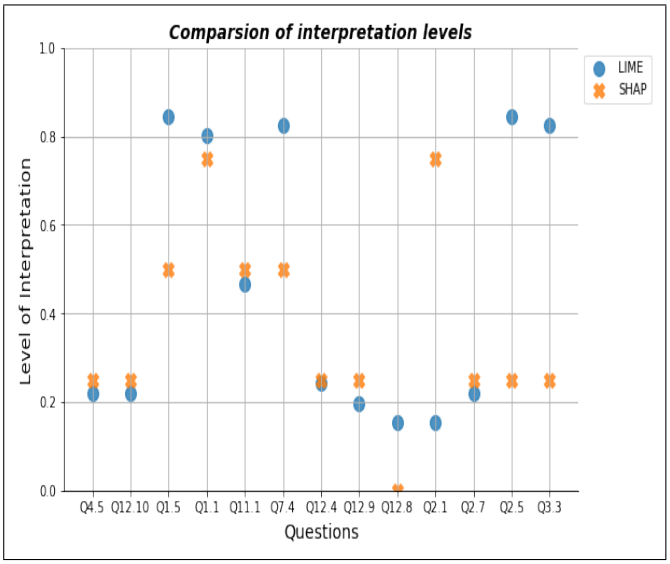

Fig. 2. Quantitative Interpretation Level comparison of LIME and SHAP for different questions.

As observed from the figure, in majority of the cases LIME results in either a comparable or higher interpretation level from its explanations. This stems from the reason that LIME's explanation being more human friendly and are more geared towards explaining the predictions to laypersons when compared to SHAP.

IV. PROPOSED SOLUTION

The algorithm developed in this work was aimed with end goal of predicting and presenting student textual responses such that a human end user's understanding of the relation between the components of the current instance and the model's prediction of the score is enhanced in a way similar to the explanations given in LIME, but at the same time also attempting to address some of the limitations of LIME. LIME's restriction of indexing and highlighting only unigrams was observed to be insufficient in promoting the understanding of the human users given the diversity of student answers. Thus including words, we aim at phrases of a set length to be included in the explanation generated by our algorithm. The second main goal that we aimed to

achieve was to identify globally important features that can also be used to explain the local instance being predicted. While former research as well as the results of our prior experiments has shown that establishing globally faithful explanations that also maintains its importance in the local context is still a challenge[1], we attempt to strike a balance between the two by actively involving the human end users in the grading process.

A. Algorithm

The strategy employed by the algorithm is mentioned in the following text:

- Starting from an unlabeled dataset, a small subset of instances are selected, and are then annotated by a human oracle. Unlike in active learning strategies where the selection of data points to be annotated is done based on the predictive uncertainty of a trained model, our selection is completely random and the annotation proceeds until the initially selected subset is exhausted.
- The labeled training set is used to learn our model (using the Naive Bayes classifier) and the ngram range of (1,3) is used to learn features for each class of decision (i.e, grades).
- Features learned from the trained model are used as baseline hints to generate explanations for the unseen data. This is done in the following way:
  1) Ngrams in the range of (1,3) is first extracted and compared against the global vocabulary of features for each class. Matching features found are set to be included as part of the explanation.
  2) Trigram phrases are given the highest priority to be highlighted in the explanation followed by bigrams. In other words, if there are features present in the form of trigrams, any possible duplications in the form of bigram and/or unigrams is eliminated. The same process is also applied for bigram features.
  3) Any trigram or bigram phrases in the unseen data that begin with the same words as the features that are a part of the global vocabulary are suggested as belonging to the same class of vocabulary and are included as part of the final explanation. In case a user disagrees with the suggestion, the feedback section of the algorithm can be used to correct and update the global vocabulary accordingly.
- Features recognized as part of an explanation are highlighted and the human oracle is allowed to suggest changes in the score as well as the explanation.
- Features suggested by the human oracle is used to update the current vocabulary of explanation features, and the change is highlighted to the human user.
- The new instance and its label is used to update the model and recompute the global feature sets for each class of grades. The global vocabulary is then updated with pool of feedbacks received upto the present instance.

Algorithm 1 Generate explanation with active feedback

Input: unlabeled dataset $\mathcal{D}$

Select n random samples $\mathcal{X} \leftarrow \{x_1, x_2 ... x_n\}$ from $\mathcal{D}$ as *training* and the rest as *test*.
for Each x_i in $\mathcal{X}$ **do**
    Obtain annotation y_i for x_i from human oracle.
end for
Preprocess the data and train model ϕ on annotated dataset.
Obtain feature sets $\mathcal{F}^{\mathcal{C}_k}$ corresponding to each class $\mathcal{C}_k$ where $k = \{0, 1, 2\}$ and each feature f_i is an ngram in range (1,3).
for Each instance x_i in *test* **do**
    Get model's decision on the grade.
    Derive ngram features in range (1,3) from instance x_i into $\mathcal{N}$.
    Obtain explanations $\mathcal{E}^{\mathcal{C}_k}$ such that a feature $e_j \in \mathcal{E}^{\mathcal{C}_k}$ iff $e_j \in \mathcal{F}^{\mathcal{C}_k}$ for the set of classes $k = \{0, 1, 2\}$.
    for Each n_i in $\mathcal{N}$ **do**
        $\mathcal{E}^{\mathcal{C}_k} \leftarrow n_i \cup \mathcal{E}^{\mathcal{C}_k}$ if n_i begins with the same word as one of the f_i in $\mathcal{F}^{\mathcal{C}_k}$ where $k = \{0, 1, 2\}$.
    end for
    for Each explanation set $\mathcal{E}^{\mathcal{C}_k}$ **do**
        for Each feature e_i in $\mathcal{E}^{\mathcal{C}_k}$ **do**
            if Feature e_i in $\mathcal{E}^{\mathcal{C}_k}$ also occurs as part of a larger ngram in $\mathcal{E}^{\mathcal{C}_k}$ **then**
                Remove ngram e_i from $\mathcal{E}^{\mathcal{C}_k}$.
            end if
        end for
    end for
    Highlight model's decision and explanation.
    Get user feedback on correctness of model prediction and update model's decision for current instance.
    Obtain user feedback $\mathcal{V}^{\mathcal{C}_k}$ on correctness of highlighted explanations for $k = \{0, 1, 2\}$.
    Update annotated dataset with $\mathcal{X} \leftarrow \mathcal{X} \cup x_i$ and using corrected decision.
    $\mathcal{E}^{\mathcal{C}_k} \leftarrow \mathcal{E}^{\mathcal{C}_k} \cap \mathcal{V}^{\mathcal{C}_k}$ for $k = \{0, 1, 2\}$.
    Display corrected explanation and update model ϕ.
    Recompute feature sets $\mathcal{F}^{\mathcal{C}_k}$ followed by $\mathcal{F}^{\mathcal{C}_k} \leftarrow \mathcal{F}^{\mathcal{C}_k} \cap \mathcal{V}^{\mathcal{C}_k}$ for $k = \{0, 1, 2\}$.
end for

B. Interface Design

The architecture for the developed GUI is shown in Figure 3. The whole dataset which is input in the form of nbgrader metadata is first extracted, parsed and converted into comma separated value (CSV) files on a per question basis directly from nbgrader metadata. Tasks such as selecting a specific question for grading, grading the initial answers, and providing feedback for the later predictions are done through the front end. The backend is responsible for preprocessing and extracting the features, learning a model, predicting and generating explanations, updating model from received feedback and finally storing and integrating the final grades and explanations into the jupyter notebooks. The GUI was

designed to be hosted and deployed as a web with the micro web framework Flask was used to route requests and responses between the HTML client and the python web server.

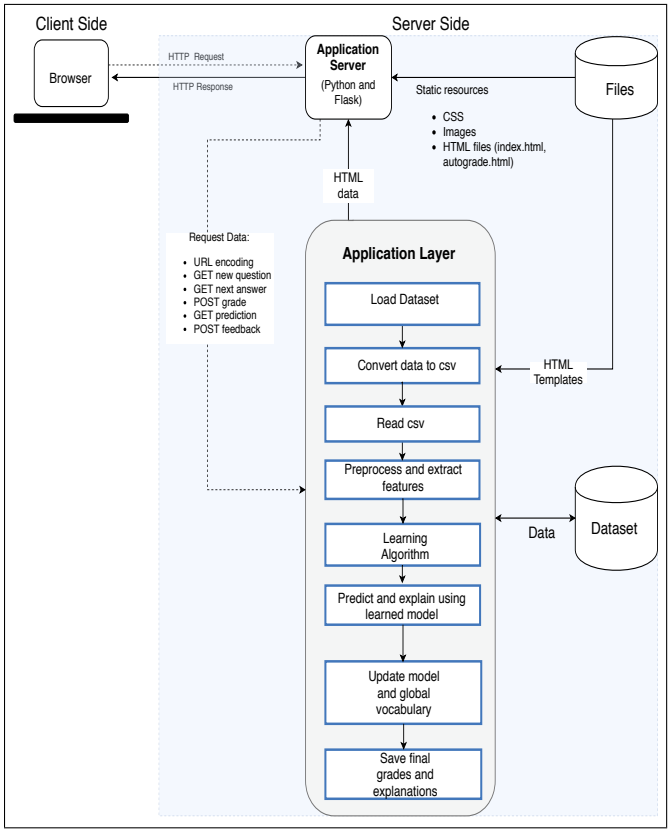

Fig. 3. Software Architecture of GUI

With the goal of having a minimalist and easy to use design for the interface, the GUI frontend was set with one radio button for each possible grade. In order to assist the human users during the initial grading, a section displaying the model answer to the current answer is provided. An example snapshot of the GUI during the initial annotation of the answers is shown in Figure 4.

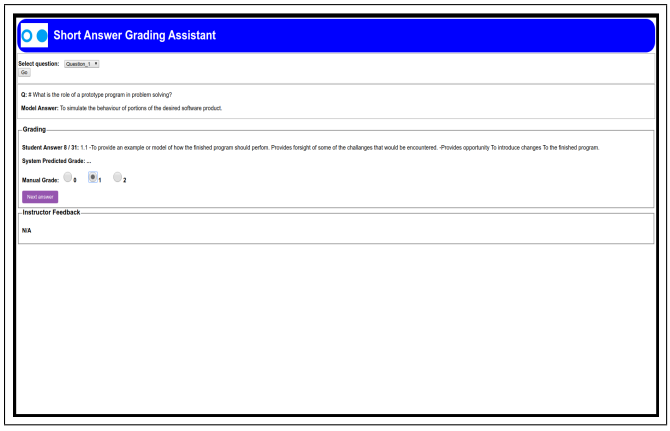

Fig. 4. GUI display during the initial annotation stage.

After a fixed set of answers are graded by the users, the tool starts assigning grades to the remaining unseen answers. Figure 5 shows the GUI display during this stage where the features contributing to the grade are highlighted in a color specific to that grade. In our implementation, a features influencing a score of 2 points are highlighted in green, those for 1 points are highlighted in yellow and red for zero points.

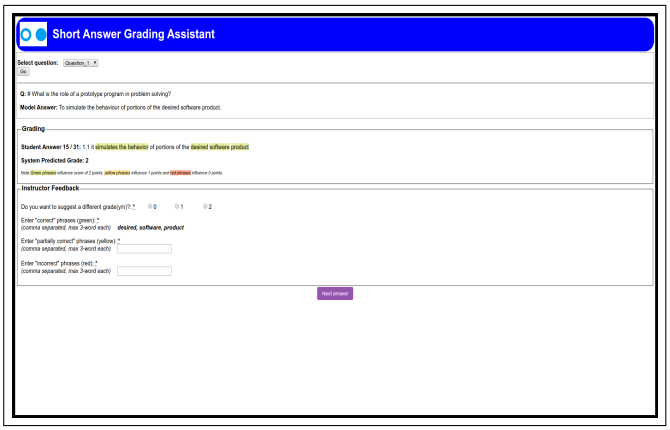

Fig. 5. GUI display during the autograding and feedback stage.

As seen in Figure 5, phrases upto three words and not just single words are highlighted as part of the explanation. In addition, the human user may provide feedback in case of a disagreement with the system predicted grade and explanation that helps to subsequently improve the grades and explantions for the new unseen answers.

The results of the quantitative evaluation done by repeating the procedure carried out in our preinvestigations, i.e, using a Likert scale of 3 values (1,2 and 3) to score the degree of the acceptance of the displayed explanation for every answer to all questions from Mohler's dataset used as case studies, and then using the SciPy librarys interpolation function to map the maximally occurring score for a question into the ranges [0,0.33], [0.34,0.66] and [0.67,1] (for 1, 2 and 3 respectively) is shown in Figure 6.

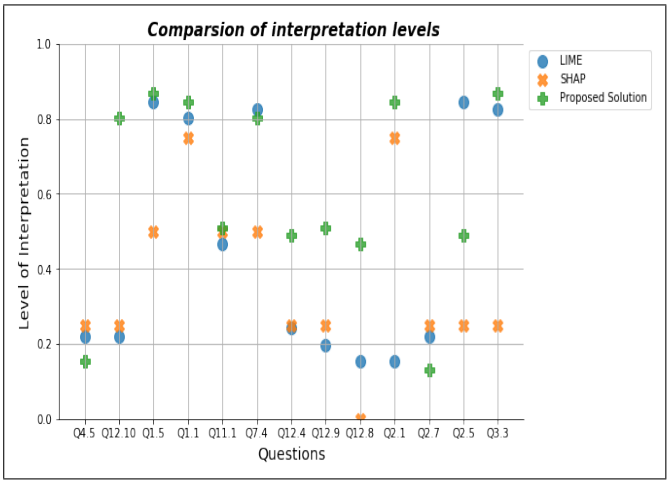

Fig. 6. Quantitative Interpretation Level comparison of LIME, SHAP and the proposed solution for different questions.

We observe that in the majority of the cases, the performance from the proposed solution in terms of the level of interpretation gained from the explanation is better than or is at least as good as that obtained from LIME and SHAP.

V. USABILITY STUDY

Since the development of the autograding assisting system is geared towards use in examinations and in-house assignments, a user evaluation was conducted at the University of Applied Sciences Bonn-Rhein-Sieg aimed at assessing the quality of the developed grading system through interacting with real users, and thereby determining the requirements and limitations that were overlooked due to the experimenter's bias. A full-time member of the University's lecturing staff and four other research associates were asked to attempt using the software to grade a number of questions from the Mohler's dataset as part of the evaluation study. Prior to the evaluation, each user was given a brief presentation on the usage of the system. Each user was then asked to grade multiple questions from the dataset. The questions selected differed in the length of the responses required from the students. The users were asked to 'think out loud' and their words and actions were video recorded. The completion of the grading of all student answers to one question was immediately accompanied by a short online questionnaire to determine whether the explanations given was useful to the users.

VI. CONCLUSIONS

....

ACKNOWLEDGMENT

....

REFERENCES

[1] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, pages 1135-1144, 2016.

[2] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems, pages 4765-4774, 2017.

[3] Christoph Molnar. Interpretable Machine Learning. https://christophm.github.io/interpretable-ml-book/, 2018. https://christophm.github.io/interpretable-ml-book/.

[4] Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin. "Anchors: High-Precision Model-Agnostic Explanations." AAAI Conference on Artificial Intelligence (AAAI), 2018

[5] David Martens and Foster Provost. Explaining data-driven document classifications. 2013.

[6] Daniel Gutierrez. Layer-wise Relevance Propagation Means More Interpretable Deep Learning. Open Data Science, 2018. URL https://opendatascience.com/layer-wise-relevance-propagationmeans-more-interpretable-deep-learning/.

[7] Ando Saabas. Interpreting random forests. http://blog.datadive.net/interpreting-random-forests/, 2014.

[8] Michael Mohler, Razvan Bunescu, and Rada 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: Human Language Technologies-Volume 1, pages 752-762. Association for Computational Linguistics, 2011.

[9] Ahmed Ezzat Magooda, Mohamed Zahran, Mohsen Rashwan, Hazem Raafat, and Magda Fayek. Vector based techniques for short answer grading. In The Twenty-Ninth International Flairs Conference, 2016.

[10] Hao-Chuan Wang, Chun-Yen Chang, and Tsai-Yen Li. Assessing creative problem-solving with automated text grading. Computers & Education, 51(4): 1450-1466, 2008.

[11] Debra T Haley, Pete Thomas, Anne De Roeck, and Marian Petre. Measuring improvement in latent semantic analysis-based marking systems: using a computer to mark questions about html. In Proceedings of the ninth Australasian conference on Computing education-Volume 66, pages 35-42. Australian Computer Society, Inc., 2007.