Application of Model Interpretability Techniques 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

....

II. RELATED WORK

....

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 [?]. 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.

B. Data Preprocessing

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:

- The model answers and student answers were tokenized into words and all words were converted into lowercase form.
- The segmented text obtained from the previous step was then subjected to stopword removal.
- Lemmatization was performed in order to obtain the base / dictionary form of a word after removing the inflectional endings in each word.

[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]Bernard D. Researcheris 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`

C. 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.

D. Benchmarking Experiments' Results and Analysis

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 $\mathcal{D}$
1: Select n random samples $\mathcal{X} \leftarrow \{x_1, x_2 ... x_n\}$ from $\mathcal{D}$ as training and the rest as test.
2: for Each $x_i$ in $\mathcal{X}$ do
3:     Obtain annotation $y_i$ for $x_i$ from human oracle.
4: end for
5: Preprocess the data and train model $\phi$ on annotated dataset.
6: 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).
7: for Each instance $x_i$ in test do
8:     Get model's decision on the grade.
9:     Derive ngram features in range (1,3) from instance $x_i$ into $\mathcal{N}$.
10:    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 $k = \{0, 1, 2\}$ .
11:    for Each $n_i$ in $\mathcal{N}$ do
12:        $\mathcal{E}^{\mathcal{C}_k} \leftarrow n_i \cup \mathcal{E}^{\mathcal{C}_k}$ if $n_i$ begins with the same word as one of the $f_i$ in $\mathcal{F}^{\mathcal{C}_k}$ where $k = \{0, 1, 2\}$.
13:    end for
14:    for Each explanation set $\mathcal{E}^{\mathcal{C}_k}$ do
15:        for Each feature $e_i$ in $\mathcal{E}^{\mathcal{C}_k}$ do
16:            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
17:                Remove ngram $e_i$ from $\mathcal{E}^{\mathcal{C}_k}$.
18:            end if
19:        end for
20:    end for
21:    Highlight model's decision and explanation.
22:    Get user feedback on correctness of model prediction and update model's decision for current instance.
23:    Obtain user feedback $\mathcal{V}^{\mathcal{C}_k}$ on correctness of highlighted explanations for $k = \{0, 1, 2\}$.
24:    Update annotated dataset with $\mathcal{X} \leftarrow \mathcal{X} \cup x_i$ and using corrected decision.
25:    $\mathcal{E}^{\mathcal{C}_k} \leftarrow \mathcal{E}^{\mathcal{C}_k} \cap \mathcal{V}^{\mathcal{C}_k}$ for $k = \{0, 1, 2\}$.
26:    Display corrected explanation and update model $\phi$.
27:    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 $k = \{0, 1, 2\}$.
28: end for
```

B. Interface Design

The architecture for the developed GUI is shown in Figure 1. The whole dataset 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.

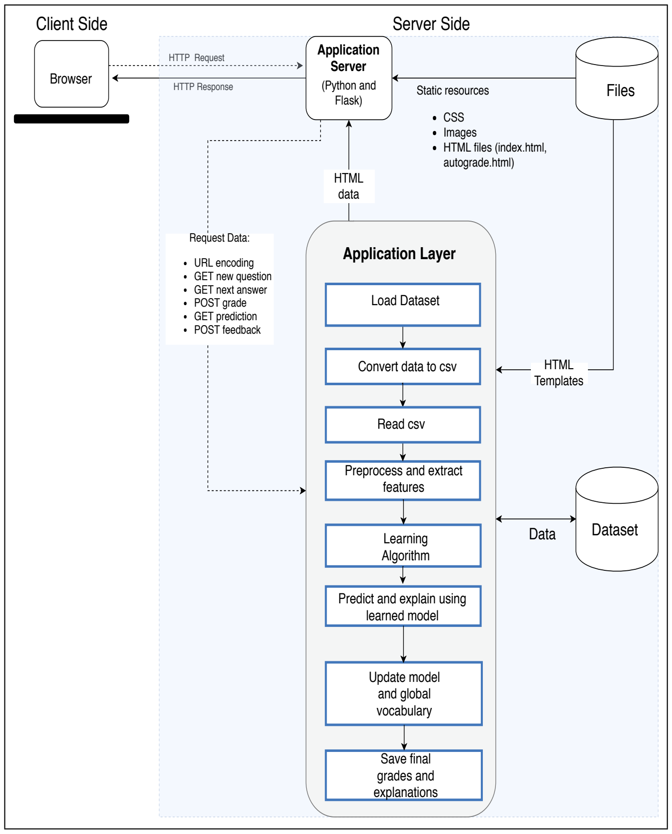

Fig. 1. Software Architecture of GUI

V. USABILITY STUDY

VI. CONCLUSIONS

....

APPENDIX

....

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.