DeepQR: Neural-Based Quality Ratings for Learnersourced Multiple-Choice Questions

Lin Ni\textsuperscript{1}, Qiming Bao, Xiaoxuan Li, Qianqian Qi, Paul Denny, Jim Warren, Michael Witbrock, Jiamou Liu\textsuperscript{2}

School of Computer Science, The University of Auckland, New Zealand
{lin\textsuperscript{1}, jiamou.liu\textsuperscript{2}}@auckland.ac.nz

Abstract

Automated question quality rating (AQQR) aims to evaluate question quality through computational means, thereby addressing emerging challenges in online learnersourced question repositories. Existing methods for AQQR rely solely on explicitly-defined criteria such as readability and word count, while not fully utilising the power of state-of-the-art deep-learning techniques. We propose DeepQR, a novel neural-network model for AQQR that is trained using multiple-choice-question (MCQ) datasets collected from PeerWise, a widely-used learnersourcing platform. Along with designing DeepQR, we investigate models based on explicitly-defined features, or semantic features, or both. We also introduce a self-attention mechanism to capture semantic correlations between MCQ components, and a contrastive-learning approach to acquire question representations using quality ratings. Extensive experiments on datasets collected from eight university-level courses illustrate that DeepQR has superior performance over six comparative models.

Introduction

Recent shifts towards online learning at scale have presented new challenges to educators, including the need to develop large repositories of content suitable for personalised learning and to find novel ways of deeply engaging students with such material (Dhawan 2020; Davis et al. 2018). Learnersourcing has recently emerged as a promising technique for addressing both of these challenges (Kim 2015). Akin to crowdsourcing, learnersourcing involves students in the generation of educational resources. In theory, students benefit from the deep engagement needed to generate relevant learning artefacts which leads to improved understanding and robust recall of information, a phenomenon known as the generation effect (Crutcher and Healy 1989). In addition, the large quantity of resources created from learnersourced activities can be used by students to support regular practice, which is known to be a highly effective learning strategy (Roediger III and Karpicke 2006; Carrier and Pashler 1992), especially when spaced over time and when feedback is provided (Kang 2016).

Despite the well-established benefits of learnersourcing for students (Moseley, Bonner, and Ibey 2016; Ebersbach, Feierabend, and Nazari 2020), evaluating and maintaining the quality of student-generated repositories is a significant challenge (Walsh et al. 2018; Moore, Nguyen, and Stamper 2021). Low quality content can dilute the value of a learnersourced repository, and negatively affect students’ perceptions of its usefulness. On the other hand, the identification of high-quality content can facilitate useful recommendations to students when practicing. Therefore, approaches for accurately assessing the quality of student-generated resources are of great interest. Involving domain experts in the evaluation process is very costly and does not scale, negating one of the key benefits of learnersourcing. A more scalable solution is to have students review and evaluate the content themselves (Darvishi, Khosravi, and Sadiq 2021). Prior research has shown that students can make similar quality judgments to experts, especially when the assessments provided by multiple students are aggregated (Abdi et al. 2021).

However, a sufficient number of students must view and evaluate each artefact before a valid assessment can be produced, which is inefficient. Computing quality assessments of content, at the moment it is produced, would benefit all learners. Such a priori assessment of quality remains a difficult yet important challenge in learnersourcing contexts.

Current learnersourcing tools support a wide variety of artefact types, including hints, subgoal-labels, programming problems and complex assignments (Mitros 2015; Kim, Miller, and Gajos 2013; Leinonen, Pirttinen, and Hellas 2020; Pirttinen et al. 2018; Denny et al. 2011). Multiple-choice questions (MCQs) are a very popular format in learnersourcing platforms, appearing in tools such as RiPPLE (Khosravi, Kitto, and Williams 2019), Quizzical (Riggs, Kang, and Rennie 2020), UpGrade (Wang et al. 2019) and PeerWise (Denny, Luxton-Reilly, and Hamer 2008). Hence, a generalisable model that can assess the quality of student-generated MCQs has the potential for a significant impact.

In this work, we explore the problem of automated question quality rating (AQQR) by developing a computational method to rate student-generated MCQs a priori. Existing measures of MCQ quality target explicitly-defined features such as cognitive complexity with respect to Bloom’s taxonomy (Bates et al. 2014), the justification of rationales (Choi, Land, and Turgeon 2005), and the feasibility of distractors (Papinczak et al. 2012; Galloway and Burns 2015). Such measures require costly and subjective manual evaluation
by experts. Recent progress in natural language processing (NLP) provides a suite of tools for extracting and analysing rich features of texts, presenting a real opportunity to enhance existing measures of quality.

**Contribution.** We propose DeepQR, a novel neural network-based model for AQQR that is trained using datasets collected from student-generated MCQ repositories. DeepQR is designed to be used in learnersourcing platforms to provide useful and immediate feedback to students and instructors. Figure 1 illustrates a potential use of DeepQR in practice. To the best of our knowledge, this is the first work that employs deep learning techniques to produce ratings of question quality. The design of DeepQR is guided by three research goals: (1) Capitalise on existing work that has identified indicative criteria for question quality – such as readability or word count – and utilise these for AQQR using tools from NLP. In particular, we investigate the extraction of correlations between various MCQ components: the question stem, the correct answer, the distractors and explanation. (2) Given the immense success of neural-based models for natural language understanding, it is natural to consider their application to the extraction of meaning from MCQs. The second research goal thus seeks to utilise rich semantic features to solve AQQR. (3) The research goals above suggest two sources of input features that are potentially useful for AQQR, namely, the explicitly-defined features (EDF) discussed in (1), and the semantic features (SF) discussed in (2). The third research goal is to explore their “interplay”, i.e., how combining EDF with SF could facilitate a superior model for AQQR.

To answer (1), we propose two complementary models: the first is an AQQR model that takes 18 EDFs as input including word counts, clarity, correctness, and readability indices. Since these features were not designed to capture relations between different question components, in the second model, we employ a self-attention mechanism that discovers semantic-based correlations of question components (SCQC). We demonstrate that enriching the input features in the first model with SCQC dramatically improves AQQR performance. To answer (2), we design a model that feeds GloVe embeddings into a transformer to produce a representation that captures the semantics of an MCQ. We demonstrate that this semantic feature-based model solves AQQR with better performance than benchmark models such as RoBERTa and SBERT which are much more costly to train, and at a comparable performance as the second model designed for (1). This demonstrates the value of semantic features (SF) in estimating question quality. To answer (3), we propose the DeepQR model by combining EDF, SCQC, and SF as discussed above. Furthermore, to improve performance, we introduce a quality-driven question embedding (QDQE) scheme, which employs contrastive learning to fine-tune GloVe embeddings and better reflect quality. This embedding is then used to enhance our model, providing a considerable performance boost. Our experiments were conducted using eight datasets (comprising 15,350 questions and more than 1,000,000 student-assigned quality ratings) from the PeerWise (Denny, Luxton-Reilly, and Hamer 2008) learnersourcing platform, collected from medicine and commercial law courses. For most courses, our model achieved accuracy in excess of 80%. We also conduct a systematic analysis to validate our design decisions and the applicability of our model.

**Related Work**

Evaluating the quality of questions has attracted significant interest in educational research. Standard metrics from classical test theory and item response theory, such as discrimination indices, have been used for decades to aid instructors in identifying poor quality questions (Brown et al. 2021; Malau-Aduli et al. 2014). However, computing such quantitative measures requires large quantities of data on student responses to items. Various qualitative indicators have also been proposed and used, such as question clarity (Choi, Land, and Turgeon 2005) and distractor-plausibility (Bates et al. 2014). Prior studies assessing such qualitative measures, however, have involved significant manual effort by experts. In learnersourcing contexts, aggregating student ratings to assess the quality of questions is scalable and agrees well with expert ratings of quality (Abdi et al. 2021; Darvishi, Khosravi, and Sadiq 2021; McQueen et al. 2014). In our own work, we use averaged student ratings as ground-truth labels when training our AQQR models.

AQQR has recently attracted the attention of the machine learning community. In particular, the inaugural NeurIPS 2020 Education Challenge included a quality prediction task for mathematics questions as Task 3 (Wang et al. 2020). Two of three of the successful entries (1. Yuto Shinahara and Daichi Takehara and 2. TAL Education Group) relied on explicitly-defined features (EDF), such as difficulty and readability, deriving a final rating using some form of average over these feature values. Such approaches are limited in the sense that they rely on ad-hoc EDFs and linear transformations which may not provide the level of robustness and flexibility required for a diverse range of questions and courses. Neural networks are able to extract distributional semantics from texts offering greater richness and versatility. Yet, to our knowledge, there has not been a systematic effort to design neural-based models to rate question quality. Our paper aims to fill this gap by investigating the value of semantic features (SF) in AQQR. We note that the question dataset published for the NeurIPS 2020 ED-
UCATION CHALLENGE is not suitable for our purpose as the questions are largely mathematical and involve diagrams.

Tasks that closely resemble AQQR – and for which neural network-based approaches have proven useful – include question difficulty prediction (QDP) and automated essay scoring (AES). QDP requires evaluation of a difficulty score for reading comprehension questions. Huang et al. (2017) approached this task using deep learning with an attention-based CNN model, and subsequent work by Qiu, Wu, and Fan (2019) incorporated a knowledge extraction aspect. Both works rely heavily on the extraction of rich SFs from a question to predict its difficulty. AES seeks to rate an article’s quality based on its content, grammar, and organization. Early AES models generally applied regression methods to a set of EDFs (Shermis and Burstein 2003). Taghipour et al. (2014) characterise suitable reader groups of a text by leveraging domain knowledge and expert judgement while the latter captures semantic features (SF) using machine learning algorithms. We speculate that both types of features could be useful for our task. (2) QDP modelling aims to construct a prediction model for the quality rating given the feature vectors. In this paper, we adopt a simple linear layer for QDP modelling. Thus the main focus of our method is on MCQ representation. See Figure 2 for an overview of these two subtasks.

**Methods**

In this section, we first describe how EDF are extracted and used in AQQR. This is then followed by a description of our transformer-based SF extraction method. Last, we present our DeepQR model which combines modules developed for both preceding parts. Overall this section will present 5 AQQR models. Fig. 3 presents an architectural overview.

**EDF-based Models**

Earlier pedagogical research have identified EDF which determine the quality of MCQ (Papinczak et al. 2012; Bates et al. 2014; Galloway and Burns 2015). We introduce two EDF-based AQQR models: EDF-Solo and EDF-Enriched. The former trains linear weights of 18 EDF computed directly from the input texts. This model echoes earlier methods, e.g., winning bids of NEURIPS 2020 EDUCATION CHALLENGE. The latter enriches the input of EDF-Solo using semantic correlations between MCQ components (SCQC), extracted by a self-attention mechanism.

**Explicitly-defined features**. Details of the selected EDF are in Table 1. The grammatical error $\gamma$ is obtained using LanguageTool (Naber 2003). The nine readability indices (DuBay 2004) characterise suitable reader groups of a text (e.g., by revealing the cognitive complexity required to make sense of the text) in different ways. A common feature of
Table 1: Explicitly defined features.

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Options</strong></td>
<td>Number of options ( n_{\text{opt}} ), word counts ( \nu_C ), where ( C ) is taken from ( {S, A, D_1, \ldots, D_4, E} )</td>
</tr>
<tr>
<td><strong>Length</strong></td>
<td>Grammatical error rate ( \gamma_i )</td>
</tr>
<tr>
<td><strong>Correctness</strong></td>
<td>Flesch reading ease, Flesch–Kincaid, ( r_{d_1}, f_{g} ), ( r_{d_1}\times d_{4} ); Semantic embedding ( r_{d_2}, 5 ); Automated readability index ( r_{d_5} ); Spache ( r_{d_7} ); Dale–Chall ( r_{d_8} ) (DuBay 2004)</td>
</tr>
<tr>
<td><strong>Readability</strong></td>
<td>Grammatical error rate ( \gamma_i )</td>
</tr>
</tbody>
</table>

The EDF-Solo model. To measure the quality of a question, classical models generally use manually-defined linear transformations on a set of EDF. In the EDF-Solo model, we also use a linear transformation (of the 18 selected EDF) but the weights for the features are trained using linear regression with MSE (mean square error) loss. See Fig. 3(a): For MCQ \( M_i \), the predicted quality rating is

\[
\hat{r}_i = \overrightarrow{w_{\text{lin}}} \cdot \overrightarrow{\text{EDF}}_i + b_{\text{lin}}
\]

where \( \overrightarrow{\text{EDF}}_i = [n_{\text{opt}}, \nu_{C}, \nu_{S}, \nu_{G}, \nu_{E}, r_{d_1}, \ldots, r_{d_8}, \nu_{d_9}] \), \( \overrightarrow{w_{\text{lin}}} \) & \( b_{\text{lin}} \) are (18-dim) trainable weights & bias, resp.

**Semantic correlation of MCQ components.** The quality of distractors of an MCQ should be assessed in the context of other components. Indeed, a “good” distractor is expected to bear certain syntactic or semantic correlations with the question stem, correct answer, and possibly other distractors. We thus design SCQC to capture these correlations; See Figure 3 (b). The input to SCQC consists of semantic embeddings of all component: \( \overrightarrow{\text{Re}} = [\overrightarrow{\text{Re}_S}, \overrightarrow{\text{Re}_A}, \ldots, \overrightarrow{\text{Re}_E}] \), where each \( \overrightarrow{\text{Re}_C} \in \mathbb{R}^{d_{\text{em}}} \) is a \( d_{\text{em}} \)-dim embedding of component \( C \in \{S, \ldots, E\} \). These embeddings are assumed to be produced by a separate algorithm (See below). SCQC utilises a self-attention mechanism to interpret correlations:

\[
\overrightarrow{\text{Co}} = \text{Attention}(\overrightarrow{\text{Re}}, \overrightarrow{\text{Re}}),
\]

where \( Q \) is a query sequence on the context sequence \( V \), \( W \in \mathbb{R}^{d_{\text{em}} \times d_{\text{em}}} \) trainable weight matrix, and the attention score matrix \( \overrightarrow{\text{Co}} \in \mathbb{R}^{7 \times 7} \) represents component-wise correlations. We then encode this matrix by a 2-layer transformer encoder. Each transformer layer contains two sub-layers: a self-attention mechanism and a feed-forward layer, each of which has a residual connection before normalisation (Vaswani et al. 2017). Thus the output of each sub-layer is \( \text{LayerNorm}(X + \text{Sublayer}(X)) \), where \( X \) is input, \( \text{LayerNorm} \) is the normalisation function, and \( \text{Sublayer} \) is either self-attention or feed-forward as described in (3) and (4), respectively. Set the query, key, and value matrices as \( Q = XW^Q, K = XW^K, V = XW^V \) where \( W^Q, W^K, W^V \) are \( d_{\text{lin}} \times d_{\text{lin}} \) trainable matrices, respectively.

\[
\text{SelfAttention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_{\text{em}}}}) V
\]

\[
\text{FeedForward}(X) = \max(0, XW_1 + b_1)W_2 + b_2
\]

where \( W_1, W_2 \) are trainable weights and \( b_1 \) and \( b_2 \) are biases. The input to the first transformer layer is the correlation matrix \( \overrightarrow{\text{Co}} \) and the next layer’s input is the output of the previous layer. The output of the 2-layer transformer is a \( 7 \times 7 \) matrix representing the encoded correlations. Finally, we compute the average attention score for each component which is fed into a linear layer to produce the SCQC output. Note that the parameters of SCQC are trained using prediction loss and thus SCQC captures in some sense the impact of each component to the question quality. We will showcase the ability of SCQC using a case study in the section “Analysis and Discussion”.

The EDF-Enriched model. See Figure 3(b). The model concatenates the normalized EDF with SCQC output before...
applying a linear layer (similar to (1)) to obtain the rating prediction. We fix the popular GloVe word embeddings (Pennington, Socher, and Manning 2014) to represent the MCQ components as inputs to the SCQC module. GloVe was trained with local as well as global statistics of a corpus and is able to capture semantic similarity using much lower dimensional vectors than other popular word embeddings.

SF-based Model

Given the success of neural networks in building rich representations in multiple NLP tasks, it is reasonable to expect that deep semantic information captured by such models may also serve the purpose of AQQR. This section presents our method to extract such semantic information. Just as for SCQC, we use a 2-layer transformer encoder. The input of the encoder is word embeddings of the MCQ components. The transformer consists of two multi-head self-attention layers (Vaswani et al. 2017). Having multiple heads allows the discovery of richer information as different heads may focus on different aspects of the data. We then average the obtained representation and apply a linear transformation to get the final SF representation.

Figure 3(d) summarises the architecture of the SF model. We again adopt GloVe as input embedding to the SF extraction module. After obtaining the SF representation, the SF model applies a final linear layer (similar to (1)) to obtain the predicted quality rating. We mention that efficiency amounts to a key advantage of our method. Indeed, it is straightforward to train (heavyweight) models such as RoBERTa and SBERT from the input corpus. Yet, we will demonstrate in the section “Result” that this does not improve performance while incurring heavier training costs.

Models Combining EDF and SF

This section presents two models that combine the EDF and SF in the hope to make the best use of the extracted features.

The Combined model. Following Figure 3(c), the model takes the normalised EDF concatenated with SCQC, which is then concatenated with the extracted SF above. The combined vector is applied a linear transformation as in the models above to produce a quality rating prediction. The model parameters of SCQC and the SF extraction module are trained using the prediction loss. The input embedding to both SCQC and the SF extraction modules are GloVe, which is obtained without quality consideration. This inspires us to fine-tune GloVe to strengthen the representations of MCQ.

Quality-driven question embedding and DeepQR model. The DeepQR model differs from Combined in that it adopts QDQE for its input instead of GloVe embeddings. QDQE builds question representations while taking into account their quality rating. For this, we adopt a (supervised) contrastive learning algorithm to fine-tune a baseline language model. Contrastive learning gained considerable interest recently as a generic representation learning framework (Chen et al. 2020; Giorgi et al. 2021). Supervised contrastive learning leverages label information in a dataset (Gunel et al. 2021). In a nutshell, QDQE fine-tunes question embeddings derived from GloVe with a one-layer transformer encoder, so as to push apart representations of MCQ with “unmatched quality ratings”, while pulling together representations of MCQ with “matching quality ratings”.

More specifically, we build a dataset \( D_{CL} \) consisting of triples of questions. Assume our training set contains MCQ \( M_1, \ldots, M_n \), sorted in ascending order of quality rating. Set \( S_L = \{M_1, \ldots, M_c\} \) and \( S_H = \{M_{n-c+1}, \ldots, M_n\} \) where \( c < n/2 \) is a fixed integer. The dataset set \( D_{CL} \) contains triples of the form \((a, pos, neg)\): For each \((a, pos) \in S_L^2 (S_H^2)\), choose a random \( n_{a,pos} \in S_L (S_H) \). Then \( D_{CL} = \{(a, pos, neg) \mid (a, pos) \in S_L^2 \cup S_H^2, neg = n_{a,pos}\} \). Note that there are in total \( 2c(c-1) \) triples.

Figure 4 shows the model for training QDQE. The model starts from \( d \)-dim GloVe embeddings of the MCQ components for questions in a \((a, pos, neg)\) triple. These embeddings are encoded by a transformer that produces for each question a \( 7 \times d \) encoded matrix. A \( d \)-dim vector is then computed by taking the mean of each column of the encoded matrix for each question in the triple. The contrastive loss is the InfoNCE function \( \mathcal{L}_{IN} \) (Oord, Li, and Vinyals 2018):

\[
\mathcal{L}_{IN} = -\log \frac{\exp(\text{sim}(a; pos)/\tau)}{\exp(\text{sim}(a; pos)/\tau)+\exp(\text{sim}(a; neg)/\tau)} \quad (5)
\]

where \( \text{sim} \) used here is cosine similarity, and \( \tau = 0.07 \) following MoCo (He et al. 2020). In this way, we hope the distance between the QDQE vectors \( \bar{a} \) and \( \bar{p} \) becomes less than the distances between \( \bar{a} \) and \( \bar{n} \), as well as between \( \bar{p} \) and \( \bar{n} \).

Experiments

Experiment Setup

Datasets. Our eight PeerWise datasets are taken from a law course and seven medicine courses (M1,4,7 are the same course of a university in different school years, similar for M2,3,5,6). Each dataset contains MCQ that are presented as in the section “Problem Formulation” where the average rating is the ground truth. To ensure the reliability, only questions receive at least 10 ratings are included. See Table 2 for the final datasets details.

Models. We test five AQQR models (EDF-Solo, EDF-Enriched, SF, Combined, DeepQR) and two benchmark models RoBERTa and SBERT. The benchmark
models are implemented using roberta-base and sentencetransformers/paraphrase-distilroberta-base-v1 from HUGGINGFACE (Wolf et al. 2019), resp. and are fine-tuned on AQQR. For these benchmarks we combine all MCQ components as a single input. We split each dataset into training, validation, test set by 8:1:1. We use a seed of 2021, and train our models using Adam optimizer (Kingma and Ba 2015) for 50 epochs from which the epoch achieves the lowest validation loss (MSE: mean square error) is chosen.

**Performance measures.** We use both MSE and ACC to measure AQQR performance. For ACC, we count a predicted rating as correct when \(|\hat{r}_i - r_i| \leq 0.25\) (recall \(r_i\) & \(\hat{r}_i\) are resp. the ground truth & predicted labels) and define ACC as the fraction of correct predictions in the test set. While ACC offers insight on the model’s ability, MSE can be seen as a more reliable metric.

**Hyper-parameters.** We set the batch size to 16 and the initial learning rate to 1e−3. For the optimizer learning rate scheduler, we set step size to 3 and gamma to 0.7 for the optimizer learning rate scheduler. We set the dropout to 0.5 for our AQQR models and to 0.1 for the two benchmarks. For QDQE, we train a model separately for each course dataset with \(c = 80\) and 20 for the train and validation dataset respectively. The hyper-parameter settings are inherited from benchmark models except the batch size is 1.

**Experiments design.** We conduct experiments in three stages to verify modules within our design. We first compare EDF-Solo with EDF-Enriched to highlight the power of SCQC, and then compare SF with RoBERTa and SBERT to showcase our SF extraction module. We last demonstrate the value of the combined models Combined and DeepQR hoping to validate the use of QDQE.

### Results

All experiments are conducted on NVIDIA 460.84 Linux Driver by Quadro RTX 8000 with 48 GB GPU memory and CUDA version 11.2. The CPU version is Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz and 16 cores. The results are shown in Table 3. We make the following observations: (1) As seen from the first two rows, EDF-Enriched made a substantial improvement from EDF-Solo across all datasets, achieving almost 80% less MSE for M7 and 43% less MSE for M6 less. This reflects the result after enriching the EDF with SCQC. (2) Among the SF-based models, SF outperforms the benchmarks of RoBERTa and SBERT in most of the cases, and performs comparably well as EDF-Enriched. This demonstrates that deep learning captures sufficient semantic information to express question quality. (3) In general, the models that combine EDF and SF achieve the best accuracy. This is despite the fact that Combined’s lead is not conclusive on some of the datasets, e.g., getting higher loss than other models in Law and M1. This is somewhat surprising as enlarging the input features does not necessarily boost the performance. Nevertheless, DeepQR scores the best performance across all datasets in terms of both MSE and ACC. The only two exceptions are the ACC scores on Law and M5 which are within 1.4% and 0.4% of the best scores respectively. In most of the datasets, DeepQR achieves 2%+ better than the next best model in terms of ACC. This demonstrates the benefit of using QDQE as the input sentence embedding.

To illustrate the computational efficiency of our transformer-based SF extraction module, Table 4 shows the average training time per epoch of the RoBERTa, SBERT and DeepQR model. In all but one dataset, DeepQR outperforms the benchmarks.

### Analysis and Discussion

**Case study on SCQC.** One benefit of our SCQC module is its explanatory ability. By visualising the self-attention matrix, we are able to observe correlations among MCQ components which are calibrated to reveal quality. Fig. 5 displays the case study whose correlation matrix is shown as a heat map. Darker blue cells indicate a higher correlation between the components. The diagram shows high correlations between the stem, and the distractor 1 “Hallucinations” with the answer “Schizophrenia”, which help to reveal question quality. This diagram hints that SCQC facilitates a question to capture meaningful insights for the model explanation.

**Comparison of rating distributions.** Fig. 6 compares the four rating distributions on M3 dataset as a case study: ground truth, EDF-Solo, SF, and DeepQR predictions. The histogram displays the number of questions whose ratings fall within different intervals. While ratings produced by EDF-Solo are too evenly distributed, and those by SF concentrate too much in one rating interval, the rating distribution obtained by DeepQR strikes a balance between the two and most resembles the ground truth.

**Identifying questions with high- (low-)quality.** A model that can detect questions with exceptionally low or high quality could be used as either a question filter (to eliminate low-quality questions) or a question recommender (to pro-
Table 3: AQQR performance (MSE & ACC(%)) of seven models on eight PeerWise datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Law</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDF-Solo</td>
<td>0.119</td>
<td>33.64</td>
<td>0.104</td>
<td>61.14</td>
<td>0.133</td>
<td>61.58</td>
<td>0.062</td>
<td>71.42</td>
</tr>
<tr>
<td>EDF-Enr.</td>
<td>0.115</td>
<td>53.91</td>
<td>0.079</td>
<td>78.29</td>
<td>0.097</td>
<td>67.55</td>
<td>0.041</td>
<td>83.74</td>
</tr>
<tr>
<td>SF</td>
<td>0.079</td>
<td>57.25</td>
<td>0.064</td>
<td>77.71</td>
<td>0.103</td>
<td>64.90</td>
<td>0.038</td>
<td>86.21</td>
</tr>
<tr>
<td>RoBERTa</td>
<td>0.117</td>
<td>54.68</td>
<td>0.064</td>
<td>78.85</td>
<td>0.109</td>
<td>62.91</td>
<td>0.042</td>
<td>86.69</td>
</tr>
<tr>
<td>SBERT</td>
<td>0.117</td>
<td>54.68</td>
<td>0.064</td>
<td>78.85</td>
<td>0.113</td>
<td>65.68</td>
<td>0.043</td>
<td>80.29</td>
</tr>
<tr>
<td>Combined</td>
<td>0.122</td>
<td>52.08</td>
<td>0.071</td>
<td>77.14</td>
<td>0.097</td>
<td>68.89</td>
<td>0.038</td>
<td>87.19</td>
</tr>
<tr>
<td>DeepQR</td>
<td>0.107</td>
<td>56.77</td>
<td>0.060</td>
<td>80.57</td>
<td>0.093</td>
<td>68.87</td>
<td>0.036</td>
<td>88.18</td>
</tr>
</tbody>
</table>

Table 4: Average training time per epoch of RoBERTa, SBERT and DeepQR (in second).

<table>
<thead>
<tr>
<th>Model</th>
<th>Law</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa</td>
<td>109.66</td>
<td>50.00</td>
<td>43.33</td>
<td>57.00</td>
<td>34.00</td>
<td>86.00</td>
<td>35.00</td>
<td>26.00</td>
</tr>
<tr>
<td>SBERT</td>
<td>55.00</td>
<td>25.00</td>
<td>22.66</td>
<td>29.00</td>
<td>18.00</td>
<td>44.00</td>
<td>18.00</td>
<td>14.00</td>
</tr>
<tr>
<td>DeepQR</td>
<td>44.33</td>
<td>27.66</td>
<td>17.66</td>
<td>28.00</td>
<td>16.33</td>
<td>40.00</td>
<td>17.33</td>
<td>12.66</td>
</tr>
</tbody>
</table>

Conclusions and Future Work

This paper investigates AQQR using tools from deep learning. We propose the DeepQR model that combines EDF and SF sources extracted by transformer networks, as well as contrastive learning-based question embeddings. Empirical results using eight PeerWise datasets validate the superior performance of DeepQR over six comparative models. Future work includes improving the model’s accuracy through better question embedding schemes and incorporating domain-specific knowledge. In addition, when training our model we aim to improve the quality of the aggregated student ratings using effective consensus approaches (Abdi et al. 2021; Darvish, Khosravi, and Sadiq 2021). Finally, we aim to evaluate the use of DeepQR in practice for recommending high-quality questions to students.
References


McQueen, H. A.; Shields, C.; Finnegan, D.; Higham, J.; and Simmen, M. 2014. PeerWise Provides Significant Academic Benefits to Biological Science Students Across Di-


