Speaker-Conditioned Hierarchical Modeling for Automated Speech Scoring

Yaman Kumar Singla\textsuperscript{1,2,3}, Avyakt Gupta\textsuperscript{1}, Shaurya Bagga\textsuperscript{1}, Changyou Chen\textsuperscript{1}, Balaji Krishnamurthy\textsuperscript{2}, Rajiv Ratn Shah\textsuperscript{1}

1. IIIT-Delhi, 2. Adobe, 3. SUNY-Buffalo
1. India, 2. India, 3. USA

ABSTRACT
Automatic Speech Scoring (ASS) is the computer-assisted evaluation of a candidate’s speaking proficiency in a language. ASS systems face many challenges like open grammar, variable pronunciations, and unstructured or semi-structured content. Recent deep learning approaches have shown some promise in this domain. However, most of these approaches focus on extracting features from a single audio, making them suffer from the lack of speaker-specific context required to model such a complex task. We propose a novel deep learning technique for non-native ASS, called speaker-conditioned hierarchical modeling. In our technique, we take advantage of the fact that oral proficiency tests rate multiple responses for a candidate. We extract context vectors from these responses and feed them as additional speaker-specific context to our network to score a particular response. We compare our technique with strong baselines and find that such modeling improves the model’s average performance by 6.92% (maximum = 12.86%, minimum = 4.51%). We further show both quantitative and qualitative insights into the importance of this additional context in solving the problem of ASS.

KEYWORDS
automated speech scoring, spontaneous speech, end-to-end, hierarchical modeling, multi-modal, interpretability, AI in education

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM ’21, November 1–5, 2021, Virtual Event, QLD, Australia
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8446-9/21/11... $15.00
https://doi.org/10.1145/3459637.3482395

1 INTRODUCTION
Automated Scoring (AS) of language is one of the most popular applications of artificial intelligence. The first such reporting goes back to 1960s for the Project Essay Grade (PEG) by Ellis Page [41]. Since then, the systems have expanded by leaps and bounds and now they grade millions of test-takers per year [3, 23, 31]. The scores given by them are used in important decisions including college admissions, visa approvals, screening interviews, and even high-school exams [3, 23, 38, 67]. Apart from its immense importance in the education and learning domain, automatic scoring also has a large market size of more than USD 110 billion, with a US market size alone of USD 17.1 billion [32, 52, 61, 65].

AS, in general, refers to the act of using computers to convert a candidate’s performance on standardized tests to some performance metric. AS systems are required to interpret and analyse the candidate’s response and make a prediction that can be used to infer the person’s ability [15]. Automatic Speech Scoring (ASS), a type of AS, assesses the speaking proficiency of a candidate in a language. ASS systems are used for a variety of reasons such as to decide admissions to universities, alleviate the workload of teachers, save time and costs associated with grading, and evaluation of online courses. For instance, a British teacher spends an average of 8 hours per calendar week scoring exams and assignments [20]. This figure is even higher for developing and low-resource countries where the teacher to student ratio is dismal [39]. ASS systems can effectively reduce this workload while still maintaining quality [29]. Due to this reason, schools in Ohio and Utah have already started automating essay scoring for their high school students and would soon require speech scoring systems [23, 38, 67]. Additionally, with the rise of people taking online courses, online platforms require ASS systems to provide immediate results and feedback to the user. On account of all these reasons, much research and development is required in ASS to better the systems which play such an important part in life-changing decisions including university admissions, visas, and job interviews.

Currently, the majority of work in deep learning based AS has been done in the Automatic Essay Scoring (AES) domain. Speech scoring is a much harder problem than essay scoring as we have to model both the content (what was spoken) and delivery (how it was spoken). We also have to deal with open grammar, unstructured or semi structured content, prosody, fluency and pronunciation variations. Moreover, it has been noted in previous studies that applying AES approaches for ASS results in significant performance drops [33] due to the mentioned issues. To deal with these challenges, most of the previous studies have used handcrafted features for scoring speech. For instance, TOEFL\textsuperscript{®} from Education Testing Service (ETS) is a well-known example which uses a rich set of handcrafted features for ASS [76]. However, handcrafted rules are unable to model higher level features such as opinion formation,
arguments, prose coherence, etc. Therefore, recently more and more approaches are moving towards end-to-end deep learning models for scoring speech responses. For instance, Chen et al. [10] combine acoustic and lexical models to perform ASS. Qian et al. [49] and Grover et al. [19] use LSTM-attention and attention fusion, respectively, to score spontaneous speech responses. Recently, research studies have also shown that incorporating speech transformers for scoring improves performance [19, 69].

Contrary to the true classroom settings, a majority of existing feature engineering and deep-learning research treats automatic speech scoring of samples of the same candidate on different prompts\(^1\) to be independent of each other [6, 74]. That is why most of the previous literature reports the performance of one model per prompt with no or minimal information sharing between two models. It is well-known that a speaker’s performance on different prompts is related. Even in classroom settings, while scoring a response, a teacher knows the subtleties of each pupil on account of her knowledge of the pupil’s performance on previous occasions [6, 74]. Therefore, conditioning models on speakers can lead to information sharing across models corresponding to different prompts and hence is expected to improve the models’ performance.

Fig. 1 presents the basic idea of speaker-prompt interaction. Response \(r_{ij}\) is determined by two factors: speaker \(s_i\) and prompt \(p_j\). Speaker \(s_i\) remains constant while prompt \(p_j\) changes for generating (answering) a response \(r_{ij}\). Therefore, a model finding the score \(o_{ij}\) for a response \(r_{ij}\) can benefit from sharing speaker \(s_i\)’s information with other prompts \((p_k \text{ s.t. } k \neq j)\). The same idea of information sharing through speaker conditioning has been explored directly or indirectly in other tasks including speaker activity detection [13], speaker diarization [50], text-to-speech [22], speech translation [36], and speaker detection [27] in the speech domain, and authorship verification [25], author-based predictive writing [24], sentiment analysis [75], and natural text generation [40] in the NLP domain. Conditioning on speakers improved the performance on all these tasks.

Building on these gaps in existing research, we propose a novel deep learning architecture which relies on speaker-conditioned hierarchical modeling, where the prompts of each user are bundled by the user-specific context. Specifically, we feed our per-prompt models with user-specific context, allowing the models to better understand the user and make a more informative prediction (§3.4). Using an exhaustive set of experiments involving 108 unique model-prompt pairs, we compare our technique with strong baselines and show the superiority of the proposed speaker-conditioned hierarchical models (§4). We also show that audio features along with text-based features can boost the model performance, as audio features capture details such as stress, hesitation, word intensity, word duration, etc. which are not directly captured by text transcripts and hence not modeled by the text based models. In §4.1 we analyze quadratic weighted kappa and mean squared error to see how well our models perform in comparison to the baselines. We analyze how conditioning on other prompts changes the predictions for high-bias examples. We also note the speaker level accuracy increase by conditioning on speaker for the proposed models. Later, in §4.2, we interpret the results to see how the speaker-conditioned hierarchical models obtain better predictions than the baseline models. For this, we analyze the information-sharing strengths across prompts (§4.2).

The main contributions of our work are summarised as follows:

- We propose two types of speaker-conditioned hierarchical models for the problem of ASS. To the best of our knowledge, this is the first time speaker conditioning is considered on a multi-prompt dataset in ASS.
- Through extensive experiments on 108 unique model-prompt pairs on a real-life non-native speaker dataset, which contains 7,198 speakers answering 6 prompts each, we show that the proposed models achieve state-of-the-art results on speech scoring. Our method increases average human-machine agreement by 6.92% (maximum = 12.86%, minimum = 4.51%) over strong baselines.
- Further, using interpretability experiments, we show how our model learns the relations between prompts and individual speaker responses via speaker-conditioned hierarchical embeddings. We find that this also increases the precision for all prompt-specific models. We also find the relative importance of audio and text in ASS.

2 RELATED WORK

Automatic Speech Scoring (ASS): In general, ASS can be of the following types: read-aloud, repeat-aloud, keyword sentence completion, structured narrative, providing an opinion, listen-speak, and conversation-based assessments [16]. Amongst them, structured narrative, providing an opinion, and conversation-based assessments are the most thorough and hence the most challenging ones [16]. In this work, we cover the first two of these three. The range of features covered by this type of assessment is highly comprehensive and include fluency, grammar, vocabulary, discourse coherence, and content, amongst others [73]. Due to the complexity of the task, earlier systems relied on simply measuring automatic

---

\(^1\)Here a prompt denotes an instance of a unique question asked to test-takers for eliciting their opinions and answers in an exam. The prompts can come from varied domains including literature, science, logic and society. The responses to a prompt indicate the creative, literary, argumentative, narrative, and scientific aptitude of candidates and are judged on a pre-determined score scale.
Table 1: Statistics of the SOPI-SLTI dataset. Difficulty measures the CEFR scale difficulty of each prompt. We report the distribution of each prompt across proficiency levels in the “Score Distribution” column.

| Prompt | Difficulty | Avg #sec. | Avg #tokens | Score Distribution |
|--------|------------|-----------|-------------|--------------------|
| 1      | B1         | 57.79     | 101.91      | A2(197)/Low B1(1286)/High B1(5715) |
| 2      | B1         | 58.82     | 111.18      | A2(400)/Low B1(2697)/High B1(4101) |
| 3      | B2         | 81.86     | 151.58      | A2(65)/Low B1(463)/High B1(3087)/Low B2(3497)/High B2(86) |
| 4      | C1         | 104.83    | 183.83      | A2(65)/LB1(463)/High B1(3087)/LB2(3497)/High B2(86) |
| 5      | C1         | 106.67    | 199.61      | A2(60)/LB1(382)/High B1(2675)/LB2(3945)/High B2(136) |
| 6      | B1         | 56.15     | 111.22      | A2(56)/Low B1(767)/High B1(6375) |

Over the years, a lot of studies have focused on extracting hand-crafted features that cover different aspects of speech such as fluency, rhythm, intonation and stress, vocabulary use, etc. [76]. However, these systems are not able to capture complex high-level features such as opinion formation, structure of prose, argument depth, etc. Studies have shown that end-to-end systems work better than feature-based systems, while also obviating the need for feature engineering.

Chen et al. [11] were the first to propose an end-to-end network that captures both lexical and acoustic cues to score samples. They used Bi-directional LSTM with attention to generate features from text, and for acoustic cues they used Kaldi ASR’s [47] outputs to obtain acoustic model posterior probabilities and word durations, and Praat [8] to obtain pitch and intensity values. They reported an improvement over the conventional method of handcrafted features [76]. Saeki et al. [51] proposed a combination of three models: lexical, acoustic and visual models to score conversation based assessments. Grover et al. [19] proposed a multimodal end-to-end system, which uses a Bi-directional Recurrent CNN and Bi-directional LSTM to encode acoustic and lexical cues from spectrograms and transcriptions, respectively. They applied attention fusion on these features to learn complex interactions between different modalities before final scoring and reported consistent empirical improvements of their system over strong baselines. Qian et al. [48] showed improvements over baselines after including prompt-level information in their models.

Automated Essay Scoring (AES): A task related to ASS is AES. It is one of the most important educational applications of natural language processing (NLP) in the education domain [38, 66, 67]. The first neural network approach was proposed by T&N [63]. They proposed a network which uses Convolutional Neural Network (CNN) to encode local context information, and a long short term memory network (LSTM) to encode long term dependencies. Dong [14] improved this by including the attention mechanism into the network by using attention pooling of CNN features instead of the simple max pooling/average pooling. Tay et al. [64] hypothesized that scoring essays can be improved by computing and using coherence scores of an essay, since coherence is an important dimension of essay quality. They modeled coherence by adding an additional layer to their network which takes in two hidden states of the LSTM network and outputs the similarity or coherence. They used these coherence values to enrich their features and boost their performance.

However, none of the approaches in both AES and ASS have looked into the possible gains after conditioning them on speakers. Our work differs from the current works as we explore speaker-conditioned hierarchical modeling, a way to enrich our feature space by providing speaker-specific cues obtained from previously trained models. It has been shown by various previous studies that speaker-conditioned models get better performance across many tasks such as text-to-speech and natural text generation [22, 40].

3 TASK AND DATASET
3.1 Dataset
In this study, we use the data collected by Second Language Testing Inc. (SLTI) while administering the Simulated Oral Proficiency Interview (SOPI) Exam for L2 English speakers [19, 58]. The SOPI exam has been operational since 1992 and has a rich research history [58, 60]. Currently, SOPI is used for screening interviews, university admissions, employee training and skill development, university and job placement, and as a test in several online courses [55]. The SOPI offers psychometric advantages in terms of reliability and validity, particularly in standardized testing situations [35].

A majority of the speakers in the released dataset are from the Philippines region. The candidates are high school graduates and above. A SOPI test-taker is presented with a question paper containing six prompts on their computer screens, and their responses for each individual item are recorded. The test taker is given approximately one minute to think and two minutes to respond to each question. To answer the questions in the form, the speaker needs to provide arguments and explanations supporting their opinion. These responses are rated by two expert raters, and in case of a disagreement on the overall score a third rater is asked to resolve the same. The prompts and the rubrics for evaluation follow the guidelines as proposed by the Central European Framework of Reference for Languages (CEFR) [9]. CEFR is an international standard for describing language ability. It describes the language proficiency on a six-point scale, from A1 for beginners to C2 for those who are proficient in the language.

The levels of questions vary from B1 to C1, and the ratings provided to the speakers are upper bounded by the level of the question. The distribution of the dataset and the relevant statistics is provided in Table 1. The dataset consists of 931 hours of speech spoken by 7,198 speakers. The test elicits argumentative, narrative, persuasive and expository skills to answer the questions. For example, the prompt 1 asks candidates to describe their work environment. A candidate gets two types of scores: prompt-level and overall (global)
We stratify our dataset based on the speaker and global score and then split the candidates in train, validation and test set in 70:10:20 ratio. This ensures that for each prompt we have no speaker intersection in the train, validation and test sets, as required by our speaker-conditioned hierarchical modeling approach. It is noteworthy that this also reflects the real deployment setting. For each prompt, we train our model and report the quadratic weighted kappa (QWK) score on the test set.

We use DeepSpeech2 [1] based ASR system with a trigram language model for transcription of the non-native responses. The ASR system is trained on approximately 1000 hours of audio sampled from CommonVoice [2] and LibriSpeech datasets [42] and further fine-tuned on approximately 22 hours of transcribed non-native spoken responses from our dataset. The dataset is further augmented with noise using AudioSet [18]. This ASR achieves a word error rate of 16.63%.

We tokenize the transcripts using the Spacy tokenizer [21] and lowercase them. Words not part of the training data vocabulary are mapped to the unknown token (UNK) and initialized to zero vector in the embedding layer. We treat the scoring of the responses as a regression problem such that the output $o_{ij}$ is the normalized score of the speaker $s_i$’s response $r_{ij}$ to a prompt $p_j$. The CEFR [9] proficiency levels (N levels) associated with the responses are normalized to the range $[0, 1]$ for training. While testing, we rescale the model output back to the original score range and measure the performance. All our experiments are done using PyTorch [44]. All the plots and hyperparameter tuning were assisted by Weights and Biases [7]. Next, we present our models that were used as feature extractors and the proposed modeling strategies.

### 3.3 Models

We experimented with several model architectures including Bi-directional LSTM (BDLSTM), Bi-directional LSTM with Attention (BDLSTM+Attn), and BERT based model. We also tried including audio features using wav2vec2.0 [4] with each text encoder model. We demonstrate each of these models with 3 modeling strategies thus showing results for 18 (6*3) different models for 6 prompts. In total, we experiment with 108 (18*6) individual model-prompt pairs. The three modeling strategies are: baseline strategy without any speaker-conditioning (Fig. 3), one stage speaker-conditioned hierarchical modeling (Fig. 2a), two stage speaker-conditioned hierarchical modeling (Fig. 2b). Before explaining the different modeling strategies in §3.4, we first give a brief overview of each of the models.

**Bi-Directional LSTM:** The Bi-directional LSTM is capable of learning long-term dependencies. A bi-directional network is chosen here to take into account information both from the past and future given the inherent nature of speech and language production. Our model configuration is similar to [11]. The words from the preprocessed transcripts are mapped to their embedding. This word embedding layer is initialized with 300-Dimensional glove...
embeddings [46] trained on Wikipedia and is optimized during training. We use a single layer of BDLSTM, and assign the hidden layer from the last time step as the context vector \( c \) for the entire response.

**BDLSTM+Attention:** The attention mechanism [68] has achieved state-of-the-art on various natural language processing tasks. It enables weighting contextual information learned during each time step, allowing the model to determine which states to pay attention to. We apply attention to the hidden states of a BDLSTM. Given the hidden states of the BDLSTM \( h \), we calculate the context vector \( c \) as follows:

\[
e_t = W_\text{at} h_t \\
\alpha_t = \text{softmax}(e_t) \\
c = \sum_{t=1}^{T} h_t \alpha_t 
\]

where \( h_t \) is the hidden representation at the \( t \)th timestep, \( W_\text{at} \) is the weight matrix for the attention layer which assigns importance scores (\( e_t \)) to each hidden state. These scores are then normalised to sum to 1 (\( \alpha_t \)) using a softmax layer. The context vector \( c \) is a convex combination of the hidden states with weights \( \alpha_t \).

**BERT:** We use the BERT Tokenizer and the BERT Base model from the huggingface library [72]. Pretrained BERT [12] based embeddings have set a new standard in many natural language processing tasks. The BERT base model has 12 multiheaded self-attention layers with a total of 110M trainable parameters. It produces a 768-dimensional feature vector for a text sequence. We fine-tune the last 7 self-attention layers and add a projection layer on top of the features extracted to obtain the context vector \( c \) for a response.

**Multi Modal Models:** We hypothesise that audio features, along with text-based features can improve the performance of models. We rely on the previous studies that show that audios capture features like stress, hesitation, rhythm, word duration, pitch, intensity, etc. which are absent in transcripts and hence text-based features do not give complete information required for scoring [53]. To extract features from audios, we use the pretrained wav2vec2.0 [4] model provided by the huggingface library [72]. Features extracted from wav2vec2.0 have shown SOTA results on various speech tasks [33].

Wav2vec2.0 takes in the raw audio sampled at 16khz, and outputs a 768 dimensional feature vector (\( \mathbf{a} \)), summarising the entire audio. We store these features (\( \mathbf{a} \)) of every audio extracted from wav2vec2.0 and apply a fully connected layer over these features to finetune them and obtain the audio context vector \( \mathbf{a} \).

\[
c^a = W^a \mathbf{a} + b^a 
\]

We then concatenate the context vector from the text encoders (\( \mathbf{c}^t \)) with the audio context vector (\( \mathbf{c}^a \)) to obtain the multimodal context vector \( c \) for a response.

\[
c = \text{concat}(c^t, c^a) 
\]

### 3.4 Modeling Strategies

We try out three modeling strategies: baseline strategy without any speaker-conditioning (Fig. 3), one stage speaker-conditioned hierarchical modeling (Fig. 2a), two stage speaker-conditioned hierarchical modeling (Fig. 2b). In the speaker-conditioned hierarchical modeling strategies, we provide other responses \( r_{ij} \) for different values of \( j \) of the same speaker \( s_i \) on other prompts \( p_j \) as context \( c_{ij} \) to the model \( \{m\} \) to assist its predictions \( o_{ij} \). Since a response \( r_{ij} \) is dependent on speaker \( s_i \) and prompt \( p_j \), we hypothesise that providing the model with a test taker’s context can be a crucial element in understanding his/her speech and making a more informed prediction.

**Baseline Strategy:** This is given by the Fig. 3 where we have 6 models (\( m_j \), \( j \in \{1,6\} \)), one for each prompt \( p_j \) \( (j \in \{1,6\}) \). The models are conditioned only on text transcripts \( \{T_{ij}\} \) or a combination of text and audio \( \{\text{concat}(T_{ij}, A_{ij})\} \). The text transcripts and audio are encoded by the text and audio encoders \( \{E_j, j \in \{1,6\}\} \) explained in the previous section (§3.3). We obtain the context vector \( c_{ij} \) from encoder \( E_j \) for the response of speaker \( s_i \) on prompt \( p_j \). We then pass this context vector through a fully connected layer to obtain the final score \( o_{ij} \).

\[
o_{ij} = W_m c_{ij} + b_m 
\]

The models are trained independently of each other using weighted MSE loss until convergence.

**One Stage Speaker-Conditioned Hierarchical Model:** As we show in Fig. 2a, in this approach, we condition each prompt \( p_j \)’s scoring model \( m_j \) on all the inputs \( \{T_{ik}, A_{jk}\} \) of prompts \( p_k \) s.t. \( k < j \). The context of a response is the context vector extracted from the encoder model \( (e.g. \text{final hidden state in BDLSTM}) \).

\[
c_{ij} = E_j(T_{ij}, A_{ij}) \quad (5)
\]

where \( c_{ij} \) is the context vector of a model on the \( j \)th prompt for the \( i \)th speaker \( s_i \). We define the new context vector \( c'_{ij} \) as:

\[
c'_{ij} = \text{concat}(c_{ij}, c_{ij}, ..., c_{ij}) \quad (6)
\]

It is the concatenation of context vectors from the previous prompts and the context vector from the current prompt \( j \) for a particular speaker \( s_i \). We then pass this new context vector \( c'_{ij} \) through a fully connected layer to obtain a score for the sample \( o_{ij} \).

\[
o_{ij} = W_m c'_{ij} + b_m 
\]

We train our models sequentially from prompt 1 to prompt 6, and store the context vector for every speaker after the training completes. While training the model on the \( j \)th prompt, the context vectors \( c_{ij_1} \) to \( c_{ij_{j-1}} \) are stored vectors obtained from previously trained models (Fig. 2a). We perform this experiment on all of our models, i.e., BDLSTM, BDLSTM+Attention, BERT, and text encoders with wav2vec2.0.

**Two Stage Speaker-Conditioned Hierarchical Modeling:** In this approach (Fig. 2b), we assist the training of our model by providing it the context of all the responses of a speaker from the baseline models as shown in Fig. 2b. As modeling non-native users is a harder problem [6, 17, 53], our model needs more user-specific context. The context of response is the context vector extracted from the baseline model \( (e.g. \text{final hidden state in BDLSTM}) \) that is then passed through the fully connected layer to score the sample.

\[
b_{ij} = E_j(T_{ij}, A_{ij}) \quad (8)
\]
Let \( b_{ij} \) be the stored context vector of a trained baseline model trained on the \( j^{th} \) prompt for the \( i^{th} \) speaker (\( s_i \)). Then the global context vector for \( s_i \) be defined as:

\[
v_i = \text{concat}(b_{i,j}, x_{i}, s_i) \tag{9}
\]

We define the new context vector \( c'_{ij} \) as:

\[
c'_{ij} = \text{concat}(v_i, c_{ij}) \tag{10}
\]

as the concatenation of the global baseline context vector \( v_i \) and the context vector from the current prompt \( c_{ij} \) for a particular speaker. We then pass this new context vector \( c'_{ij} \) through a fully connected layer to obtain a score for the sample \( o_{ij} \):

\[
o_{ij} = W_m c'_{ij} + b_m \tag{11}
\]

We perform this experiment on all of our models, i.e., BDLSTM, BDLSTM+Attention, BERT, and text encoders with wav2vec2.0 (Fig. 2b).

### 3.5 Training

We use the weighted mean square loss as our loss function defined as:

\[
W\text{MSE}(y_{true}, y_{pred}) = \frac{1}{N} \sum_{i=1}^{N} (y^i_{true} - y^i_{pred})^2 w_{y_{true}} \tag{12}
\]

where \( y^i_{true}, y^i_{pred} \) are the scaled ground truth, and predicted grade, respectively. \( w_{y_{true}} \) is the weight of the ground truth class. We use the weighted mean squared loss due to class imbalance in the training data as shown in Table 1. We use the Adam [26] optimizer to minimize our loss over the training data. To prevent overfitting, we train the model with early stopping on the weighted validation MSE loss. We also use the ReduceLROnPlateau scheduler as provided in PyTorch [44] to reduce the learning rate when the weighted validation MSE loss does not improve.

### 3.6 Evaluation Metric

Similar to previous studies [19, 29, 30, 43, 49, 69, 70], we have used the quadratic weighted kappa (QWK) as our evaluation metric. It measures the level of agreement between two sets of ratings. The metric outputs values between 0 (random agreement between ratings) and 1 (complete agreement between ratings). In cases where the agreement is less than expected by chance, the score can be negative too. To calculate the QWK score, firstly we calculate the confusion matrix \( O \) of size \( C \times C \), where \( C \) is the total number of classes. \( O_{ij} \) represents the number of samples with true label \( i \) and predicted label \( j \). The matrix is normalised to have sum equal 1. We calculate the weight matrix \( W \) of size \( C \times C \), where \( W_{ij} = (i-j)^2/(C-1)^2 \). The weight matrix penalises predictions that are further away from their ground truth more harshly than the ones that are closer to the ground truth. Then we create the histogram matrix of expected grades \( E \), which is calculated by taking the outer product of histograms of ground truth and predicted labels. We normalise this matrix such that the sum equals 1. The QWK is calculated as:

\[
QWK = 1 - \left( \sum_{i,j} W_{ij}O_{ij} \right) / \left( \sum_{i,j} W_{ij}E_{ij} \right) \tag{13}
\]
Apart from QWK metric, following the recommendations of other studies [30, 34], we analyze the results more comprehensively by calculating speaker-level accuracy, performance on near decision-boundary samples and high-biased samples. In order to gain more intuition, we perform attributions over both modalities and also show information sharing strength across prompts. The results of all these analyses are presented in the next section.

4 RESULTS
Here we present the quantitative and qualitative results for the different models and modeling strategies.

4.1 Quantitative Analysis

QWK Score: In Table 2, we compare the two proposed speaker-conditioned hierarchical modeling strategies with baseline strategies on various models. It can be inferred from the table that speaker-conditioned hierarchical modeling consistently improves performance across all the models on all prompts. This result supports our hypothesis that speaker-specific cues assist the model in making better informed predictions. In table 2, we also report the average QWK for each model which we define as the average of promptwise QWK values for a model. The mean improvement in average QWK across all models is 4.97% (maximum = 7.32%, minimum = 2.05%) and 6.92% (maximum = 12.86%, minimum = 4.51%) in one stage and two stage hierarchical models compared to their baseline counterparts.

Similar to average QWK, we also compute average MSE for each model. As compared to the baseline, we observe a mean decrease in average MSE by 10.03% and 15.21% in one stage and two stage hierarchical models, respectively. We see that the two stage speaker-conditioned hierarchical BERT model performs better than the other models in three out of six prompts. For the other three, the one stage speaker-conditioned hierarchical BERT model performs better. This result strongly indicates that conditioning on speakers has improved the results.

Table 2 also contains results for experiments with text and audio features. Similar trends as text-only models are also observed here. First, we see an improvement in performance of 61.11% multimodal models as compared to the text-only models. Our one stage and two stage speaker-conditioned hierarchical BERT model beats the BERT text-only model. This shows that our hypothesis of audio-specific features not being captured within text transcriptions was true and including wav2vec2.0’s features helped improve the performance. Speaker-conditioning the models further improves the performance by 5.9% as compared to the multi-modal baseline models.

Speaker Level Accuracy: Since our hypothesis is that including speaker-level information from previous models should improve the performance of subsequent models, we also show improvements in speaker-level accuracy. For this, we first compare the mean number of prompts correctly classified for each speaker by the three modeling strategies, i.e., $\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{6} I(\hat{y}_{true,i,j} = \hat{y}_{pred,i,j})$ where $\hat{y}_{true,i,j}$, $\hat{y}_{true,i,j}$ are the predicted output and the ground truth for $i^{th}$ speaker on $j^{th}$ prompt. We found that the mean number of correctly classified prompts for one stage hierarchical and two stage hierarchical models are 4.018 and 4.014, respectively, which is 7.4% better than the baseline mean of 3.740.

To strengthen our claims further for speaker level accuracy, we compare the number of speakers for whom responses on at least $k$ prompts are predicted correctly for different values of $k$. We present the results in Fig. 4a, where we see that both the one-stage hierarchical model and the two stage hierarchical model outperform the baseline for all the values of $k$. In Fig. 4b, we can see the significance of this improvement as compared to the baseline model. As the value of $k$ increases, the relative improvements become more significant. This observation suggests that speaker-specific cues used in speaker-conditioned modeling help our model increase speaker level accuracy.

High Bias Samples: We set out to see if conditioning on speakers makes the models avoid errors where the model’s predictions are far from the ground truth. Based on our hypothesis, we expect that given more speaker level information, the models should be able to perform better on such samples. Consequently, we analyze samples which were predicted incorrectly by more than 2 score levels. As compared to baseline models, we observe up to 60% and 55% reduction in high-bias errors in one stage
and two stage hierarchical models, respectively. Results obtained on prompt 4 are presented in Fig. 5. This shows that giving more speaker-specific context reduces errors on high-bias samples.

**Bias Due To Performance On Other Prompts:** Since with speaker conditioning, we are giving more speaker context to the models, models might become unfairly biased towards a speaker’s previous (good or bad) performance in some prior prompt. For instance, if a speaker performs poorly (well) on some prior prompt but better (poor) on the current prompt, the speaker should not be unfairly penalized (awarded) because of his previous performance.

We analyse the models’ predictions for those samples on which the speakers have a high predicted score on prompt \(i - 1\) and low ground-truth scores on prompt \(i\). For the other case (where predicted score on prompt \(i - 1\) is low and ground truth on prompt \(i\) is high), the number of samples were too less to obtain any statistical inference. We find that one stage and two stage hierarchical models tend to give a higher score than the ground truth score of the speakers have a high predicted score on prompt \(i - 1\) and low ground-truth scores on prompt \(i\). Moreover, the average percentage of correctly predicted samples in the one stage and two stage hierarchical models were higher by 10.30% (maximum = 4.43%), respectively, than the corresponding baseline models. The results for this are shown in Fig. 7.

**Integrated Gradients:** In Section 4.1, we see an improvement in the QWK score. In order to further probe how our technique helps in improving model performance, we use an attribution method to analyze models’ predictions with respect to their inputs. We employ the method of integrated gradients [62] for this purpose:

Given an input \(x\) and a baseline \(b\), the integrated gradient along the \(i^{th}\) dimension is defined as:

\[
IG_i(x, b) = (x_i - b_i) \int_{a=0}^{1} \frac{\partial F(x + a(x - b))}{\partial x_i} \, da \tag{14}
\]

where \(\frac{\partial F(x)}{\partial y_i}\) represents the gradient of \(F\) along the \(i^{th}\) dimension of \(x\). We choose the baseline as empty input (all 0s) since an empty speech sample should get a score of 0 as per the scoring rubrics. For the first experiment on prompt-wise attribution, we took prompt-wise response embeddings as the dimension of attribution. For the second experiment on finding the contribution of modalities, we took text and audio modalities across all prompts, as the two dimensions for finding attributions.

Integrated gradients have been successfully used in the past for various NLP tasks [37, 43, 54]. To calculate these attributions, we used the Captum [28] library and aggregated all attributions of the test set to retain a global view of the importance of different embeddings.

**Prompt-wise Attribution:** To assess information sharing strength across prompts, we compute prompt-wise attributions on the two stage hierarchical models. The results for this are shown in Fig. 7. Here we observe that the most crucial embedding in predicting each prompt is the embedding from the prompt itself, which is expected since this embedding contains most of the context related to the response of that prompt. Additionally, we can see that there are many lighter patches in Fig. 7. These patches represent the importance of embeddings from other prompts in the current model’s predictions.

As can be seen in the heatmap, embeddings from prompts 1 and 2

---

*Defined as an input containing absence of cause for the output of a model; also called neutral input [54, 62].*
were the most important and contributed significantly in predicting other prompts. This validates our hypothesis that providing a model with rich user specific context via hierarchical modeling can be useful for improving its performance.

**5 CONCLUSION**

In this paper, we have investigated end-to-end deep learning based models for the automated speech scoring task for L2 English speakers. We show that speaker-conditioned hierarchical models perform better than their baseline counterparts. Results indicate that providing the model with additional speaker-specific context helps the model in better understanding the speaker and making a more informed prediction. These models are more precise and less biased than their baseline counterparts. We also show that including audio based features increases models’ expressivity and generalisability, which improves the overall performance. In future work, we wish to explore the effectiveness of our technique on AES, off-topic response detection, and individual speech components such as dysfluency and pronunciation.

**Acknowledgements:** Yaman Kumar is the recipient of Google PhD fellowship and would like to thank Google for supporting him. We would also like to thank Second Language Testing Inc. (SLTI) for providing us with data. Rajiv Ratn Shah is partly supported by the Infosys Center for AI and Center for Design and New Media at IIIT-Delhi and would like to thank them.
REFERENCES

[1] Darío Amodei, Sundaram Ananthanarayan, Rishita Anandh, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qingtian Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Enric Elsen, Jesse Engel, Weisun Fang, Linxi Fan, Christopher Fougner, Lian Gao, Caixia Gong, Aven Hannun, Tony Han, Lappi Vaino Johannes, Bing Jia, Cai Ji, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangpeng Li, Dongpeng Ma, Shazan Narang, Anwei Ng, Shervil Orzay, Yiping Peng, Ryan Prenger, Sheng Quan, Zongfang Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Smet, Arupom Srinath, Haiyunan Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijuan Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dami Yagotomina, Bin Yuan, Jun Zhan, and Zhenyao Zhu. 2016. Deep Speech 2: End-to-end Speech Recognition in English and Mandarin. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (New York, NY, USA) (ICML ’16). JMLR.org, 173–182.

[2] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Greg Weber. 2020. Common Voice: A Massively-Multilingual Speech Corpus. In LREC.

[3] Educational Testing Association et al. et al. 2014. A snapshot of the individuals who took the GRE revised general test.

[4] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. CoRR abs/2006.11477 (2020). arXiv:2006.11477 https://arxiv.org/abs/2006.11477

[5] Pakhi Bambaw, Manraj Singh Grover, Yaman Kumar Singla, Rajiv Rattn Shah, Paramesh Vaare, and Mika Hama. To appear in 2021. Automated Speech Scoring System Under The Lens: Evaluating and interpreting the linguistic cues for language proficiency. International Journal of Artificial Intelligence in Education (To appear in 2021).

[6] Isaac I Bejar. 2017. Threats to score meaning in automated scoring. Validation of score meaning for the next generation of assessments: The use of response processes (2017), 75–84.

[7] Lukas Biewald. 2020. Experiment Tracking with Weights and Biases. https://www.wandb.com.

[8] Paul Boersma and David Weenink. 2009. Praat: doing phonetics by computer (Version 5.1.13). http://www.praat.org

[9] Peter Broeder and Waldemar Martyniuk. 2008. Annotation and Automatic Scoring of Students’ Essays. In Proceedings of the Language Resources and Evaluation Conference (LREC 2008, Marrakech, Morocco), Long Papers.

[10] Dario Amodei, Sundaram Ananthanarayan, Rishita Anubhai, Jingliang Bai, Eric Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dami Yagotomina, Bin Yuan, Jun Zhan, and Zhenyao Zhu. 2016. Deep Speech 2: End-to-end Speech Recognition in English and Mandarin. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (New York, NY, USA) (ICML ’16). JMLR.org, 173–182.

[11] John Highton, Sarah Leonardi, Arifa Choudhury, Neil Richards, David Owen, and Nicholas Sofroniou. 2017. Teacher workload survey 2016. (2017).

[12] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. (2017). To appear.

[13] Ben-Nun Hsu, Yu Zhang, Jie R. Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, et al. 2018. Hierarchical generative modeling for controllable speech synthesis. arXiv preprint arXiv:1810.07217 (2018).

[14] Thomas B. Fordham Institute. 2020. Ohio Public School Students. https://www. ohioiyouthnumbers.com/

[15] Shaoxiong Ji, Shuru Pan, Guodong Long, Xue Li, Jing Jiang, and Zi Huang. 2019. Learning private neural language modeling with attentive aggregation. In 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, 1–8.

[16] Milton King and Paul Cook. 2020. Authorship Verification with Personalized Language Models. In International Conference on Text, Speech, and Dialogue. Springer, 248–256.

[17] Miederer P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980 cite arxiv:1412.6980 Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[18] David Klusicek, Jiri Navratil, Douglas A Reynolds, and Joseph P Campbell. 2003. Conditional pronunciation modeling in speaker detection. In 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings (ICASSP’03), Vol. 4. IEEE, IV–804.

[19] Muzaffer Koikkiläinen, Vivek Mehta, Miguel Martin, Edward Wang, Bilal Alsalakh, Jonathan Reynolds, Alexander Melnikov, Natalia Klushkina, Carlos Araya, Sip Yen, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for PyTorch. arXiv:2009.07896 [cs.LG]

[20] Yaman Kumar, Swati Aggarwal, Debanjan Mahata, Rajiv Rattn Shah, Ponnurangam Kumaraguru, and Roger Zimmermann. 2019. Get IT Scored Using AuToAS—An Automated System for Scoring Short Answers. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33. 9662–9669.

[21] Yaman Kumar, Mehar Bhatia, Anushka Kahra, Jessy Juny Li, Jin, and Rajiv Rattn Shah. 2020. Calling out bluff: Attack ing the robustness of automated scoring systems with simple adversarial testing. arXiv preprint arXiv:2007.08796 (2020).

[22] Geoffrey T Laflair and Burr Settles. 2019. Duolingo English test: Technical manual. Retrieved April 28 (2019), 2020.

[23] Thi Le. 2020. Testing & Educational Support in the US. https://my.biworld.com/us/en/industry/61171/key-statistics.

[24] Anastasia Loukina and Aoife Cahill. 2016. Automated scoring across different modalities. In Proceedings of the 11th workshop on innovative use of NLP for building educational applications. 130–135.

[25] Nitin Madnani and Aoife Cahill. 2018. Automated scoring: Beyond natural language processing. In Proceedings of the 27th International Conference on Computational Linguistics, 1099–1109.

[26] Margaret Malone. 2000. Simulated Oral Proficiency Interviews: Recent Developments. ERIC Digest. (2000).

[27] Arya D McCarthy, Lieliz Puzon, and Juan Pino. 2020. SkinAugment: auto-encoding speaker conversions for automatic speech translation. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE: 7924–7928.

[28] Pamoud Kausik Mudraclarka, Ankur Taly, Mukund Sundararajan, and Kedar Rangam Kumaraguru, and Roger Zimmermann. 2019. Get IT Scored Using AuToAS—An Automated System for Scoring Short Answers. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33. 9662–9669.

[29] Patrick D’Oonnell. 2020. Computers are now grading essays on Ohio’s state tests. https://www.cleveland.com/metro/2018/03/computers_are_now_grading_ essays_on_ohios_state_tests_your_ch.html.

[30] OECD. 2014. Indicator D4: How much time do teachers spend teaching?

[31] Shereen Oraby, Lena Reed, Shubhangi Tandon, TS Sharath, Stephanie Lukin, and Marilyn Walker. 2018. Controlling personality-based stylistic variation with neural natural language generators. arXiv preprint arXiv:1805.08352 (2018).

[32] Ellis B Page. 1967. Statistical and linguistic strategies in the computer's scoring programs. In COLING 1967 Volume 1: Conference Internationale Sur Le Traitement Automatique Des Langues.

[33] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. 5206–5210. https://doi.org/10.1109/ICASSP.2015.7178964

[34] Nitin Madnani and Aoife Cahill. 2016. Automated scoring: Beyond natural language processing. In Proceedings of the 27th International Conference on Computational Linguistics, 1099–1109.

[35] Shaoxiong Ji, Shuru Pan, Guodong Long, Xue Li, Jing Jiang, and Zi Huang. 2019. Learning private neural language modeling with attentive aggregation. In 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, 1–8.
