ISSN: 2710-0170

Developing and Testing Audio Data Processing Modules in Python to Connect to and Data Be Scored by ASS Cloud Server

Xiaoqin Shi

College of Languages and Culture, Northwest A&F University, Yangling, Shaanxi, 712100, China

*Corresponding Author: 1135014997@qq.com

Abstract

Automatic Speech Scoring (ASS) system developed on a basis of automatic speech recognition (ASR) technology is a powerful computer-assistant tool for oral test scoring. However, due to the limits of high equipment costs and high-tech operating costs of a local ASS, ASS cloud services have become the first choice of most oral English teachers and learners. The purpose of this paper is to develop and test modules in Python to preprocess the audio data, connect to the cloud server, and convert JSON data format into common Excel form. 1056 pieces of audio data were collected from test-takers' read-aloud task of CEST-4 (College English Speaking Test band 4)) and six variables (i.e., "pronunciation", "fluency", "integrity", "speed", "duration", and "overall") were defined. After analyzing the data of the test results, it is found that the oral test score is mostly affected by the "pronunciation" and "integrity", and the accuracy of pronunciation is the strongest predictor of oral performance. The modules and functions are helpful for teachers and students to use in daily oral test/practice, and these modules can also be employed in other second language oral test scored by ASS cloud sever, like oral Chinese test. Our results can provide reference and guidance for future oral research and teaching.

Keywords

ASS; Python; Cloud Server; Audio Data Processing; JSON.

1. Introduction

Nowadays, with the development of artificial intelligence (AI) technology, AI has been profoundly involved in many fields, such as technology, medical care, business, education, etc. ChatGPT has also set off a hot wave in education field. Meanwhile, automatic speech recognition (ASR) has become a main way to replace manual input with high accuracy [1, 2]. Based on the ASR technology, automatic speech scoring (ASS) systems are a powerful auxiliary tool for oral test scoring [3-5]. It greatly reduces the need of human resources in organization [6-7] and effective scoring of oral examinations, and enhances the reliability and objectivity of scoring results [8-10]. For these reasons, ASS seems to be popular among EFL (English as a second/foreign language) teachers and learners for oral English tests. The ASS cloud service which provides on-line speech soring, has been attractive for users. It makes speech digital technology research and development companies compete each other on providing quality technology and ASS cloud services. Among well-known on-line ASSs are Ordinate [11] and SpeechRater [12]. A large number of EFL learners from around the world take the VEPT or TOEFL iBT (both online tests are scored by the ASS cloud server) to test their spoken English proficiency. Researchers also confirm that ASS cloud services play a role for improving individual oral English performance [13-14]. It is very convenient for a user to upload his/her speech to the ASS cloud service platform and get feedbacks from the server, but there are still

ISSN: 2710-0170

unanswered questions and aspects to be improved. Can a common use clearly understand returned results which are in JSON format? If he/she does not know about JSON format, which is a way to help him/her? How does a teacher do it for scoring more than 1000 pieces of audio data? Does he/she need upload thousands of students' audio and get the returned results from the sever one by one? Obviously, teachers or some researchers need a new way (or instrument) to score audio data in batches by employing an ASS cloud server. Our study will focus on developing modules in Python to help teachers or users to easily read and analyze scored data by an ASS cloud server in batches.

2. Literature Review

2.1. Related Work of ASS

The first speech scoring system was developed by Bernstein et al. in 1989, and was officially used in English spoken tests in 1990 [15]. At that time, ASS system was to score the pronunciation of the test task, and mainly to score the pronunciation of native English. Based on feature extraction and pattern matching techniques, ASS systems also used to sore the non-native speakers' pronunciation by calculating the similarity of learners' speech input with a reference template and give a score [16]. Moreover, rule-based methods were used in ASS to convert speech to text and score by comparing the differences between students' answers and standard reference answers [17].

With the rise of statistical machine translation and speech recognition technologies, ASR began to employ statistical modeling methods such as Hidden Markov Models (HMMs) and Gaussian Mixture Models (GMMs) to improve accurately identify [14.17]. At that time, the ASS mainly scores read-aloud task or read-repeat tasks [18].

This ASS relies on ASR (automatic speech recognition) and detects errors on already fixed set of text [22], and scoring indexes refer to fluency, prosody, intonation, stress and vocabulary use, etc. [23]. For example, the two most widely used ASS systems are Ordinate and SpeechRater. The former developed by Pearson company was mainly used in Versant series oral examination [24]. The latter was developed by the American Educational Examination Service Educational Testing Service and used in TOEFL online simulations of oral exam training.

In decades, scholars have mainly focused on two aspects of an ASS system. One is on ASS scoring model and process, and the other is about the consistency of human and machine [12, 25]. For instance, when comparing Ordinate and SpeechRater scoring model, Jiang (2021) and Luo (2014) [26, 27] pointed out that the Ordinate score was reflected in four components: pronunciation, fluency, completeness and vocabulary, while SpeechRater's mainly scores audio data on speech speed, speech flow, pronunciation accuracy, lexical diversity, grammatical accuracy. Chen et al, (2018) [12] explained that the scoring process of SpeechRater 5.0 based on acoustic model, language model and lexicon Hirai (2021) [6] elaborated the moderate correspondence between ASS and raters in retelling task. Bernstein and Cheng (2007) [28] employed Ordinate to score 159 speeches from candidates in oral test. And they declared ASS has high consistency (r = .75-.94) with raters on fluency and accuracy of non-native speeches about read-aloud tasks. After comparing the two ASS cloud servers, Sun (2021) [29] stresses that human-machine (Ordinate) scoring is more than 80% consistent, while SpeechRater's consistency for non-native language is only 66%. So, these technologies are suitable for TOEFL oral simulation training, but not very suitable for high-stake tests.

ASS systems in China have also developed rapidly, and have been applied to some large oral examinations. For example, Li et al. (2008) [30] took the read-aloud task in the college English oral examination as the research object, selected 836 piece of audio data of test takers, took pronunciation correctness, word repetition, speech speed, pause and average length as the indexes, and examined the consistency of man-machine scores, and got the consistency

ISSN: 2710-0170

coefficient of r = 0. 713. Gong et al. (2009) [31] examined the consistency of 4010 sentence data in the test task of "following reading", and obtained the result of r = 0. 827. Lv (2015) [32] used 8 pronunciation accuracy and 18 fluency indexes, to explore the accuracy of ASS on oral English read-aloud task, and he got that man-machine consistency coefficient was r = 0. 878. The reliability and validity of ASS in the fluency and accuracy of the evaluation of EFL oral ability were confirmed by these researchers. However, they adopted local ASS systems and collected their data returned from the ASS by costing much more labor work.

In recent years, deep learning has made important breakthroughs in natural language processing and speech recognition. Deep learning-based methods, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNNs), are widely used in ASS tasks to improve the accuracy and performance of speech scoring, as well as sematic analysis, content analysis, and emotion analysis [19-21].

In short, according to the different types of oral test tasks, ASS extracts scoring feature parameters, which can be roughly divided into phonetic features and content features. The former includes pronunciation, intonation, pitch, rhythm, speech speed, pause, prosody, fluency etc., while the latter includes lexical relevance, syntax, text complexity, grammatical accuracy, collocation, discourse, language accuracy and content relevance etc. In our study, we focus on the phonetic features in read-aloud task.

Most ASS systems or cloud servers share same principles with slightly different algorithms or scoring dimensions [25-27]. The score dimensions of the ASS in China (taking Youdao ASS cloud server as an example) are mainly pronunciation accuracy, speech fluency, sentence/word integrity, and an overall score for users' oral performance in read-aloud task. The pronunciation accuracy includes phonemes, words, stress, and intonation. Accuracy of intonation mainly refers to whether tone is clear and correct. As for fluency, it mainly refers to speech speed, pause, and duration. The audio duration is obtained by subtracting the start time from the end time of the audio. On the integrity of words and sentences, ASS mainly examines the missing, omission, and irrelevant addition of words. Moreover, based on acoustic models, language models and lexicon, ASS scores each index after generation of different scoring algorithms, and gives an overall score for takers' performances.

2.2. Workflow of ASS

Despite slight differences in their internal algorithms, ASS cloud servers share similar working procedures for scoring, and consist of a speech recognition engine, a backstage confirmation system, evaluation models, a training database, and an evaluation interface. The ASS workflow architecture is shown in Figure 1. As can be seen from Figure 1, a user uploads his/her audio file and the reference text through the evaluation interface. The speech recognition decodes and calculates audios, translates speech through ASR technology, and force alignment with the given criteria by backstage configuration system. Backstage configuration system divides the reference texts into separate words or annotates phonemes. It also provides alignment criteria for the speech evaluation engine. Training database prepares an appropriate amount of corpus for off-line training to form an evaluation model before the evaluation engine works. The speech evaluation model provides the benchmark for the engine to perform decode computation and process.

There are three main models in the ASS system (i.e., acoustic, language, and lexicon models). The function of the acoustic model is to integrate acoustic and phonological techniques, extract acoustic features from input data, and calculate the probability between the corresponding phonemes, (sounds can be converted into phonemes). Acoustic models can be optimized through training a corpus, which requires a large amount of audio data. The function of the language model is to integrate grammar and lexical knowledge, and calculate the probability of collocations in sentences. Lexicon is a pronunciation dictionary in English, which serves as the

ISSN: 2710-0170

correspondence between phonetic symbols and words. With the goal of locating the corresponding word based on the phonemes recognized by the acoustic model, the lexicon bridges between the acoustic model and the language model. Through the synergy of the above three models, the audio is decoded and transcribed into texts, and then the texts are forced to align and the resulting data are used to calculate multidimensional evaluation scores and feedback.

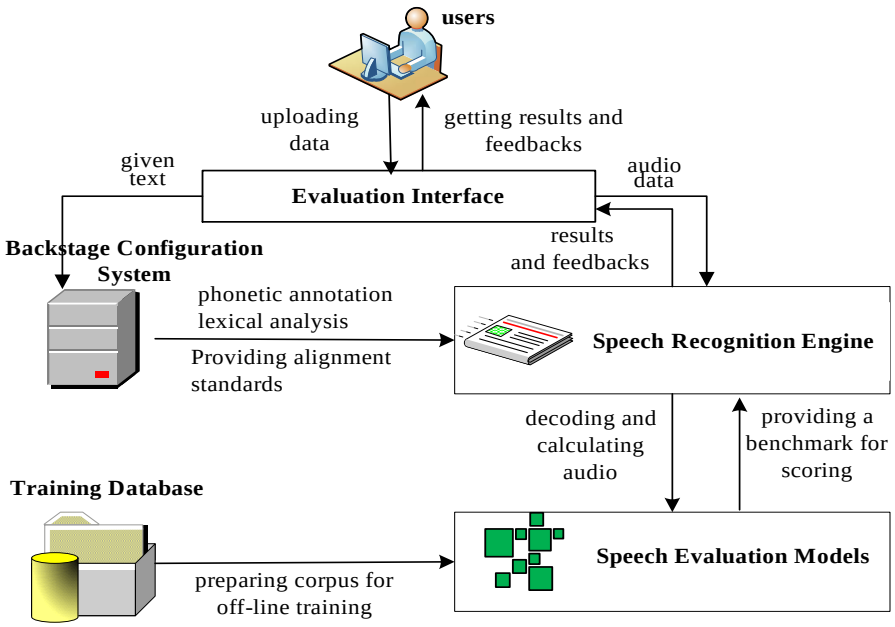

Fig. 1 The architecture of workflow of ASS

2.3. Research Questions

Although scholars have confirmed the validity and reliability of ASS in oral examination, especially in read-aloud task of oral tests, there are little research on how to adopt an ASS cloud server to evaluate audio data in batch, and make returned results easily readable for teachers or researchers in Chinese colleges. This study is going to fill the gap for addressing following research questions (RQ):

(1) How can users employ ASS could service to assess their EFL oral performance?

(2) How can users easily and clearly understand the data returned from the ASS cloud server?

(3) What can the model be built between scoring indexes to predict Chinese EFL learners' oral performance in read-aloud tasks?

3. Methods

3.1. Instrument

As aforementioned, ASS system can evaluate test-takes' EFL oral performance effectively, and users have preferences for ASS cloud services due to its high-performance cost ratio. But almost all ASS cloud services need users to preprocess their speeches for matching the server's requirements and develop a suitable evaluation interface to connect the server. In our study, we chose Youdao cloud service, owning to its affordable service cost and easily readable API (Application Programming Interface) introduction. Based on the introduction document of Youdao ASS cloud service, the authors chose Python for developing five modules and chose Pycharm to run and test Python projects. Python and Pycharm are simply introduced as follows.

Python is a programming language created and released in 1991 by Guido van Rossum. It uses a concise syntax and clear code structure, making the program easy to read and write and has

ISSN: 2710-0170

a wide range of third-party libraries and modules, providing a large number of ready-made solutions to help developers quickly implement various functional requirements. Its concise, easy-to-read, and extensible syntax is widely used in a variety of applications, including software development, data analysis, artificial intelligence, and network programming. Based on these features, it becomes one of the preferred programming languages for many developers.

PyCharm is a professional Python integrated development environment (IDE) developed by JetBrains. It provides a range of features to support Python development and is widely used for the development, debugging and management of Python projects. Therefore, Pycharm is popular among Python fans.

3.2. Developing the Modules

In our study, there are several requirements of Youdao ASS cloud service for users' audio files as follows: 1) The file must be in WAV format, 2) the audio sample rate must be 16000 Hz and mono channel, and 3) audio length cannot be over 1 minute for online data transmission easily and reducing the server response delay. Therefore, we should develop modules to preprocess our original audio for meeting Youdao ASS cloud server requirements. In addition, connecting interface/function and conversion function of returned data format should also be developed.

The main steps of developing these functions are: 1) Users' audios can be preprocessed by data preprocessing modules; 2) Preprocessed data connects to Youdao cloud server through connecting interface; 3) After the server scores data, the scored result could be returned and its JSON format is converted into Excel form; 4) Users can analyze scored results according to the Excel table data. Figure 2 shows the process of data exchange between a client and the server.

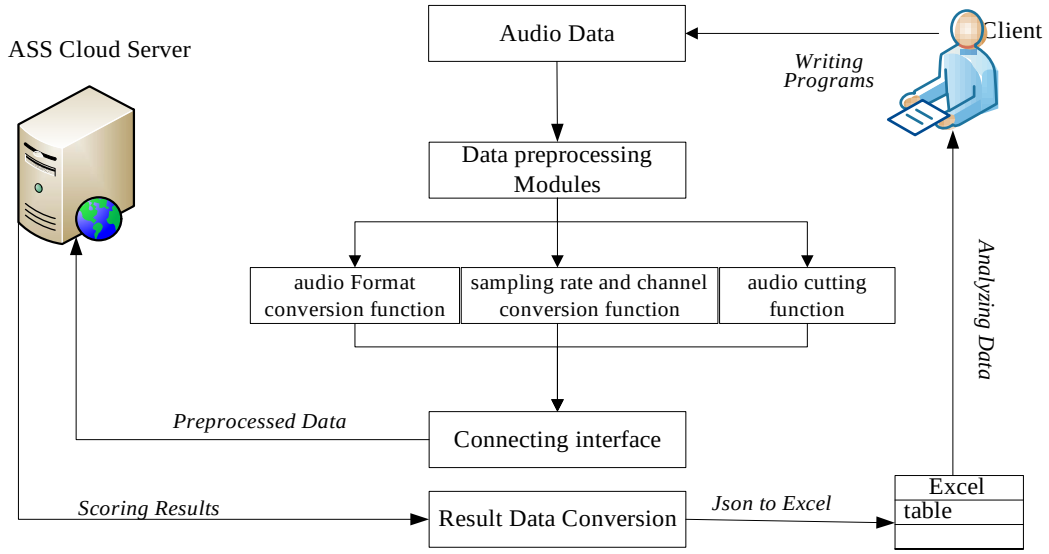

Fig. 2 The client and server data exchange schematic diagram

Currently, the main process of every developed function is shown as follows, as well as code explanations.

(1) Audio Format Conversion Function, files_convert_wav (parameter1, parameter2): This function converts the audio data in MP3 to WAV format. Parameter1 in parentheses refers to the original file path, and parameter2 is the saving the WAV file path. There are many useful tool packages in Python, and the functions in packages can be directly applied to Python programming. In the audio tool package, "pydub", there are two functions including "AudioSegment" and "export"; the former extracts the original audio data file, and the latter generates output format files; the key codes formats are as follows:

ISSN: 2710-0170

```
“def files_convert_wav(file_path, wave_path):
  files_list = os.listdir(file_path)
  print(str(files_list))
  num = 1
  for name in files_list:
    print(name)
    print(file_path + name)
    song = AudioSegment.from_file(file_path + name, format='mp3')
    d_path = wave_path + "F2020610061_" + str(num) + ".wav"
    num = num + 1
    song.export(d_path, format='wav')
  os.chdir(wave_path)
  return num”
```

In the above codes, “file_path + name” is the original file path and name and “d _ path” is the saved file path. The word “format” is the default parameter and the string after the equal sign is defined by the programmer.

(2) The author used “files_convert_wav ( )” to convert 1056 MP3 audio files into WAV files.

The conversion function of audio sampling rate and channel is “formatSpeech (SRC, dst, inrate = 48000, outrate = 16000, inchannels = 2, outchannels = 1)”. In the “formatSpeech” function, “SRC” indicates the source file, “dst” indicates the converted file, “inrate” refers to the sampling rate of the source file, “outrate” is the converted file rate. “inchannel” indicates the audio channel of the source file, and “outchannel” is the converted audio channel. According to the requirements of ASS cloud service, the original audio sampling rate in this paper is 48000, which should be converted to 16000, and the original audio channel number is 2, which should be converted to 1. In addition, “wave” and “audioop” modules sould be imported in Python programming environment. The former one is to get some information about the channel, sampling rate, and frame of the source audio. The latter one is to rewrite audio spectrum information. Specific key programming codes are shown as follows:

```
“def formatSpeech (src, dst, inrate = 48000, outrate = 16000, inchannels = 2, outchannels = 1):
  try:
    s_read = wave.open(src, 'r')
    s_write = wave.open(dst, 'w')
  except:
    print('Failed to open files!')
    return False
  n_frames = s_read.getnframes()
  data = s_read.readframes(n_frames)
  try:
    converted = audioop.ratecv(data, 2, inchannels, inrate, outrate, None)
    if outchannels == 1:
      converted = audioop.tomono(converted[0], 2, 1, 0)
  except:
    print('Failed to downsample wav')
    return False
  try:
    s_write.setparams((outchannels, 2, outrate, 0, 'NONE', 'Uncompressed'))
```

ISSN: 2710-0170

```
        s_write.writeframes(converted)
    except:
        print('Failed to write wav')
        return False
        s_write.close()
 return True"
```

(3) Audio cutting function, "splitSpeech ( )". Youdao ASS cloud service requires the length of every speech to be less than 1 minute, and in oral tests, speech time is often more than 1 minute, and this requires the source of audio files to be cut without damage to the continuity of source audios, so codes of the "splitSpeech ( )" function in Python are written to achieve audio cutting. The toolkits used in this function are the "AudioSegment" and "Make_Chunks modules" in "pydub". The main programming codes for this cutting function is shown as follows.

```
Def splitSpeech(wavepath):
    myaudio = AudioSegment.from_file(wavepath, "wav")
    chunk_length_ms = 60000  # pydub calculates in millisecond
    length = len(myaudio)
    chunks = make_chunks(myaudio, chunk_length_ms)
    # Export all of the individual chunks as wav files
    for i, chunk in enumerate(chunks):
        chunk_name = wavepath+"{0}.wav".format(i)
        print("exporting", chunk_name)
        chunk.export(chunk_name, format="wav")
```

Table 1. Codes and explanation of the required part by Youdao ASS cloud server

Key codes	Description
data[ ] = { }	Initialize dictionary data type
data['text'] = test_txt	reference text
c ur_time = str(int(time. time()))	Get the time of the system
data['curtime'] = cur_time	Pass time information to the field of data dictionary data
salt = str(uuid. uuid1())	Get Unique General Identification Code
signStr = APP_KEY + truncate(q) + salt + cur_time + APP_SECRET	Get User Digital Signature
sign = encrypt(signStr)	Encrypted digital signature
data['appKey'] = APP_KEY	Get Secret Key
data['q'] = q	Base64 encoded string of the assessed audio file
data['salt'] = salt	Send unique generic identification code to dictionary data field
data['sign'] = sign	Signature via sha256 (application)
data['signType'] = "v2"	Signature type is fixed value v 2
data['langType'] = lang_type	Source language (Chinese or English)
data['rate'] = sample_rate	Audio sampling rate
data['format'] = 'wav'	Audio format
data['channel'] = nchannels	Audio channel
data['type'] = 1	Upload type, only support base64 upload, fill in fixed value 1

(4) Conneting to the ASS server function: Connection (parameter). After the three audio file preprocessing functions, the obtained audio file can meet the requirements of the server. Next, write a interface that links the client to the server, the connect function. This function consists of three parts. The first part role is to determine whether the audio file meets the server's required format. The modules about WAV file detection are wave.open (), wave.getframerate(),

ISSN: 2710-0170

wave.getframechannels (), which in turn open the audio file, read the file sample rate and the number of channel. The second part is the transmission of dictionary data type, which is the main required part by the server. The key codes and explanations of this part are in Table 1.

The third part is to send a link request and save the returned result in the dictionary data type using the following codes (every piece of code is explained following "#"):

Data2 = bytes (urllib.parse. urlencode (data), encoding='utf-8') # data is converted to byte stream.

Response = urllib.request.urlopen (YOUDAO_URL, data = data2) # requests server processing data.

Str_response=response.read( ) decode ('utf-8') # Stores server evaluation results in variables

Return str_response # returns the variable that holds the result.

The whole connetion function codes are as follows:

```
def connection (file_name):
    audio_file_path = wav_path + file_name
    print(audio_file_path)
    lang_type = 'en'
    extension = audio_file_path [audio_file_path.rindex('.') + 1:]
    if extension != 'wav':
        print('not being supported file type')
        sys.exit(1)
    wav_info = wave.open(audio_file_path, 'rb')
    print(wav_info)
    sample_rate = wav_info.getframerate()
    print(sample_rate)
    nchannels = wav_info.getnchannels()
    print(nchannels)
    wav_info.close()
    with open(audio_file_path, 'rb') as file_wav:
        q = base64.b64encode(file_wav.read()).decode('utf-8')

    data = {}
    data ['text'] = test_txt
    curtime = str(int(time.time()))
    data['curtime'] = curtime
    salt = str(uuid.uuid1())
    signStr = APP_KEY + truncate(q) + salt + curtime + APP_SECRET
    sign = encrypt(signStr)
    data['appKey'] = APP_KEY
    data['q'] = q
    data['salt'] = salt
    data['sign'] = sign
    data['signType'] = "v2"
    data['langType'] = lang_type
    data['rate'] = sample_rate
    data['format'] = 'wav'
```

ISSN: 2710-0170

```
    data['channel'] = nchannels
    data['type'] = 1
    response = do_request(data)
    str_respnse = response.read().decode('utf-8')
    return JSON.loads(str_respnser)
```

In the the above codes, where the bold part is two functions and their explainations are shown in Table 1. They are defined as follows.

```
def encrypt(signStr):
    hash_algorithm = hashlib.sha256()
    hash_algorithm.update(signStr.encode('utf-8'))
    return hash_algorithm.hexdigest()

def do_request(data):
    data2 = bytes(urllib.parse.urlencode(data), encoding='utf-8')
    response = urllib.request.urlopen(YOUDAO_URL, data=data2)
    return response
```

The connetion function builts a brige between a client and Youdao cloud server to realize data exchange.

(5) Returned data conversion function, “JSON_to_excel (parameter)”. Most of the results from cloud servers are output in JSON (JavaScript Object Notation), a commonly used data exchange format. Because JSON data can be parsed and processed quickly by manchine, JSON can widely used in data exchange, API interface, and confiugure files. In ASS cloud serverice, developers are fond of JSON in data exchange and API interface. However, JSON has many variables, including sentences, words, phonemes, and pronunciation scores for each letter, starting time, ending time, fluency, completeneness, total score, reference text, etc. If there were a few sentences, the returned results would be possibly with several pieces of A4 paper. For someone who doesn't know much about JSON data, the returned result looks like a bunch of obscure codes. A piece of code about JSON is shown as follows:

```
{
    "integrity": 100,    //sentence integrity
    "refText": "have a good day ",    //the text of speech
    "pronunciation": 77.592987,    //overall accuracy of pronunciation
    "start": 0.030000,    //speech start time (second unit)
    "words": [{    //word information list
        "pronunciation": 99.078613,    //word pronunciation accuracy
        "start": 0.030000,    //word start time
        "end": 0.210000,    //word end time
        "word": "have",,    //word text
        "phonemes": [{    // phonemes information list
        "stress_ref": false,    // The vowel stress reference (i.e., standard stress), if true, means that the reference answers that the vowel should be stressed and that the consonant is meaningless
            "pronunciation": 99.712486,    //phoneme accuracy value
            "stress_detect": false,      // In a word, the user pronounces the phonetic symbol as unstressed
            "phoneme": "h",    //the letter of the beginning word
            "start": 0.030000,    //letter start time
```

ISSN: 2710-0170

"end": 0.090000, //letter end time

"judge": true, // To determine whether the phonetic symbols are correct, true means correct pronunciation, false means wrong pronunciation.

"calibration": "h", // To give a prompt for correct pronunciation of phonetic symbol.

"prominence": 75.887459 // Degree of stress, the more likely the current phonetic symbol is to be stressed, score interval is [0 100].

},

words after "//" are the explanatory part, and the original returned result has no such explanations.

Table 2. Main codes of conversion from JSON to Excel

Code	Description
fx = xlwt. Workbook(encoding='utf-8',style_compression = 0)	Create a workbook
sheet = fx. add_sheet("my_test", cell_overwrite_ok=True)	Create a sheet form
for m in range(len(cols)):	Start of loop statement
sheet. write(0, m, cols [m])	Initialize sheet-header
excel_data = JSON. loads(data_resp)	Encapsulates the server return results in JSON format
del excel_data["refText"]	Delete audio corresponding text in JSON format
del excel_data["words"]	Delete detailed scoring words and phonemes in JSON format
sheets. write(num, 0, file_name)	In the excel table, write the audio file name in the first column
k = 1 *sheets. write(num, k, str(excel_data ["pronunciation"]))*	Write "pronunciation" value in the second column
k = k + 1 *sheets. write(num, k, str(excel_data ["fluency"]))*	Write flow data in the third column
k = k + 1 *sheets. write(num, k, str(excel_data ["speed"]))*	Data for write rate in the fourth column
k = k + 1 *sheets. write(num, k, str(excel_data ["integrity"]))*	Write integrity data in the fifth column
k = k + 1 *sheets. write(num, k, str(excel_data ["overall"]))*	Write Total Data in Column 6
k = k + 1 *sheets. write(num, k, str(excel_data ["start"]))*	Write Start Time Data in column 7
k = k + 1 *sheets. write(num, k, str(excel_data ["end"]))*	Write End Time Data in column 8
fx. save(txt_path + 'my_test. xls')	Save the results in the my_test. xls file

Above codes show merely a measure of the pronunciation of "h" in the word "have" in the sentence "have a good day". As can be seen from codes, the return result JSON format is very complex and lengthy. Therefore, the author wrote the "JSON _ to _ excel ( )" function to convert returned results into Excel form. Because most users are interested in the scoring parameters of spoken English, like total pronunciation, total fluency, word/sentence integrity, speech speed, audio duration, and total score. In other words, we just want to get the scores of holistic evaluation indexes (i.e., "pronunciation", "fluency", "integration", "speed", "start", "end" and "overall") from the returned results. "Duration" is gotten by subtracting the "start" value from the "end" value. The specific code for the "JSON _ to _ excel ()" function is shown in Table 2.

ISSN: 2710-0170

4. Results

4.1. RQ1 and RQ2: How Can Users Employ ASS Could Services to Assess Their EFL Oral Performance? How Can Users Easily and Clearly Understand the Data Returned from an ASS Cloud Server?

After developing the above five functions, the authors collected the audio sample of CEST-4 (College English Spoken Test Band 4) from a university in the first half of 2022, and selected the first part (read aloud) of test-takers' speeches as research samples.

Based on the above-developed three functions (i.e. audio format conversion, audio sampling rate and channel conversion, and audio cutting function), the collected data were preprocessed to match the requirements of the cloud server. And the preprocessed audio data can be evaluated by Youdao cloud serve through connection interface. And then, the server returned the scoring results in JSON format. When getting the results which had been converted into Excel data by "JSON_to_excel (parameter)" function, users can easily read and analyze the returned data. Authors chose 20 pieces data from collected sample to test our functions. And then, asked two raters who have more than ten-year experience on rating CEST to evaluate the same data. The consistency of rater-ASS is r = .83, which verified that our five functions work well. Therefore, we used ASS cloud service to evaluate 1056 pieces of audio data. The data of the test results are shown in the figure 3, which can be easily understood by researchers or teachers, as well as students. Teachers can analyze the students 'spoken English performances, and make appropriate changes to teaching methods and contents in time to improve the students' spoken language proficiency.

1	id	pronunciation	fluency	speed	integrity	overall	start	end
2	F2020610061_1.wav	89.48	94.74	132.53	100.00	91.59	0.66	5.61
3	F2020610061_10.wav	80.51	90.26	148.65	100.00	84.41	0.72	5.13
4	F2020610061_100.wav	85.00	88.63	133.33	92.25	86.45	0.63	5.10
5	F2020610061_1000.wav	85.90	92.95	121.55	100.00	88.72	0.03	5.43
6	F2020610061_1001.wav	89.89	94.95	112.24	100.00	91.92	0.96	6.81
7	F2020610061_1002.wav	91.57	95.79	153.85	100.00	93.26	0.63	4.89
8	F2020610061_1003.wav	86.48	93.24	137.50	100.00	89.18	0.30	5.07
9	F2020610061_1004.wav	79.95	86.10	121.21	92.25	82.41	0.33	5.25
10	F2020610061_1005.wav	90.06	95.03	131.74	100.00	92.05	0.48	5.46
11	F2020610061_1006.wav	89.81	94.90	126.44	100.00	91.84	0.39	5.58
12	F2020610061_1007.wav	89.49	94.74	130.18	100.00	91.59	0.48	5.52
13	F2020610061_1008.wav	65.50	68.48	105.96	71.47	66.70	0.06	4.56
14	F2020610061_1009.wav	90.81	95.41	126.44	100.00	92.65	0.09	5.28
15	F2020610061_101.wav	90.23	95.12	137.50	100.00	92.19	1.08	5.85
16	F2020610061_1010.wav	90.07	95.04	146.67	100.00	92.06	1.50	5.97
17	F2020610061_1011.wav	92.82	92.54	141.84	92.25	92.71	2.25	6.45
18	F2020610061_1012.wav	75.91	83.48	131.58	91.05	78.94	2.58	7.11
19	F2020610061_1013.wav	87.97	93.99	126.44	100.00	90.38	0.42	5.61
20	F2020610061_1014.wav	95.70	97.85	125.71	100.00	96.56	0.63	5.85
21	F2020610061_1015.wav	93.48	96.74	138.36	100.00	94.78	0.75	5.49
22	F2020610061_1016.wav	83.09	91.54	148.65	100.00	86.47	0.45	4.86
23	F2020610061_1017.wav	87.32	89.79	152.67	92.25	88.31	0.48	4.38
24	F2020610061_1018.wav	84.66	88.45	85.11	92.25	86.18	0.36	7.38
25	F2020610061_1019.wav	79.63	89.82	141.94	100.00	83.70	0.30	4.92

Fig. 3 JSON Data Conversion to Excel

Fig. 3 shows the results of the first 24 piece of test results, in which the values are the overall scores of each audio file on each index. "Overall" is the total score given by ASS, equivalent to the total score of a test paper.

The developed functions can really help teachers to test students' oral English in class and quickly get the scored results by ASS cloud servers. Moreover, all students' scores are clearly shown in Excel files, which makes teachers conveniently further analyze.

4.2. RQ3: What Can the Model Be Built between Scoring Indexes to Predict EFL Learners' Oral Performance in Read-aloud Tasks?

Based on the excel form data obtained, the author makes a general analysis on the characteristics of our samples. 43 of 1,056 pieces of audio data were given a 0 value of

ISSN: 2710-0170

"pronunciation". Considering the packet loss phenomenon of the data transmitted by the network, the author compared the ID number again to find out the original audio. After performing manual comparison, the author found out that speeches are voiceless, and then discard 43 pieces of data. Then the Z value is calculated, and the audio data of Z < -2 and Z > 2 are dropped, which indicate that the speech time is too short or too long. For example, if the student ends with only one or two words, or if the student speaks only a few words for a long time, silence will be too long. According to the Z value, 48 pieces of data were removed, and 965 pieces of data were subjected to subsequent analyses.

We set "overall" as the dependent variable, and try to find its relationships with independent variables (i. e., "pronunciation", "fluency", "integrity", "speed", and "duration").

After conducting Pearson Correlation Analyses, we got the results shown in Table 3.

Table 3. The correlation between "overall" with other independent variables, respectively

Variables	Statistic	Pronunciation	Fluency	Speed	Integrity	Duration	Overall
Overall	Pearson Correlation	.987**	.949**	0.033	.586**	.200**	1
	Sig	.000	.000	.307	<.001	.000	

As can been seen from Table 3, 1) there is strong correlation between "overall" and "pronunciation" or "fluency" and correlation coefficients are r=0. 87 and r=0. 83, respectively, 2) "overall" has moderate correlation with "integrity" and the correlation coefficient is r=0. 54; and 3) there is no statistically significant correlation between "overall" and "speed" (r = .033, p = .307 > .05), and "duration" (r = .2).

Currently, it is still unclear what relationships exist between "overall" and each of the independent variable with which "overall" has significant correlation. The study conducted regression analyses. The study selected the histogram of residuals of "overall", DW values, and Collinearity Diagnostics options using SPSS 26 to test 1) if the residuals follow a normal distribution, 2) if independent variables are auto-correlative, and 3) if independent variables are collinear. The results are shown in Table 4.

Table 4. The Linear Model between "overall" and the predictors

	Model Summary			Coefficients (dependent variable = "overall")				
					UN-SC	Sig.	Collinearity Diagnostics	
Model method	**R**	**R^2**	**Durbin-Watson**	**Predictors**	B		T	VIF
Step-wise	1.0	1.0	1.986	**(Constant)**	1.10E-5	<.001		
				pronunciation	0.6	.000	0.217	4.616
				fluency	0.4	.000	0.217	4.616
				Integrity (Excluded)		<.001	8.76E-13	1.14E+12
	The Linear Equation: "overall" = 0.6 * "pronunciation" + 0.4 * "fluency" [a]							
Enter	1.0	1.0	2.008	**(Constant)**	9.89E-6	.000		
				Pronunciation	0.80	.000	0.8	1.25
				Integrity	0.20	.000	0.8	1.25
				Fluency (Excluded)			2.37E-13	4.21E+12
	The Linear Equation: "overall" = 0.8 * "pronunciation" + 0.2 * "integrity" [b]							

Note: UN-SC = Unstandardized Coefficients; T = Tolerance. VIF = Variance Inflation Factor.

a: The linear model built when "overall" is the independent variable and "pronunciation" and "fluency" are predictors. The model method is "Stepwise"

b: The linear model built when "overall" is the independent variable and "pronunciation" and "integrity" are predictors. The model method is "Enter."

ISSN: 2710-0170

The histogram of the standard residuals is normal distribution which is a prerequisite of regression analysis, and DW values is close to 2 which indicated variables are not auto-correlative, when its T (toleration) value <0.1 or VIF (variance inflation fact) value >10, the variable is collinear with other independent variables, which should be excluded from predictors. Based on above tests and when the model method is "stepwise", "pronunciation" and "fluency" are predictors of "overall" definitely. Similarly, when the model method is "enter," "overall" is completely influenced by "pronunciation" and "integrity" for sure due to both R2 = 1.0, and the significant linear equations are as follows:

"overall" = 0. 6*"pronunciation" + 0. 4*"fluency" ① (Model method = "stepwise ")

"overall" = 0. 8*"pronunciation"+ 0. 2 *"integrity" ② (Model method = "enter").

Because "pronunciation" and "fluency" have strong collinearity (by Pearson Correlation Analysis, r = 0.885), so the equation ① is not stable. But "pronunciation" and "integrity" have no strong correlation and collinearity. Equation ② is more stable. The equation was verified from the data on the figure 3. The value from the equation is almost in accordance with "overall" scored by the ASS server. So the equation ② is representative.

5. Discussion

Scholars have done a lot of researches about the reliability of ASS systems in oral tests, showing that ASS is a powerful tool for oral test scoring on some objective tasks. At present, the ASS cloud service is popular on the network. Its working principle is based on the acoustic model, language model and lexicon, and ASR technology [29]. However, a number of ASS cloud services require writing modules to complete data exchange between the clients and the ASS cloud server. For example, users need to write speech preprocessing module to meet the requirements of the cloud server on audio data, write interfaces or functions to link the server to realize data exchange between the client and the server. ASS cloud service companies provide their API documents in various programming languages (e. g., C#, Java, and Python), and clients develop their own modules/functions with their favorite programming languages for employing the ASS cloud service.

Based on above reasons, the authors developed five functions for employing the Youdao ASS cloud service in Pycharm which is an IDE (integrity development environment) of Python. The first three of functions for preprocessing collected sample to meet the requirements of the server, such as audio format conversion, sampling rate and channel conversion, and audio cutting functions. The fourth one is connection interface for uploading audio data and getting scored results. The last one is to convert the long and obscure JSON format results into a simple and straightforward Excel format for analysis.

Python provides many useful tool packages which can be imported and directly used in developer projects. In our projects, we imported "pydub" package, which contains many audio process modules, such as "AudioSegment", "make_chuncks", and "silence" to preprocess original audios. Of course, some developers use "silence" to cut the audio, set the threshold value of "silence". We use "make_chunks" to cut the audio because it automatically cut the audio at silence location near the set audio length (1 minute). In our study, most test takers' audios are less than 1minute in read aloud task and the minority of test takers' audios take longer than one minute. Moreover, pause and silence are metrics of fluency dimension of CAF (complexity, Accuracy, Fluency, three dimensions) which widely used for evaluating L2 learners' oral performance by researchers. Therefore, it is not suitable to set a threshold value of silence in our study. We used "wave" package to read the audio information, such as sample rate, channel, and framerate. Similarly, there are Timestamp, Weblink, JSON parse, Excel parse and some other packages to realize the connection and result conversion in our Python project. Some

ISSN: 2710-0170

Python fans adopt “pandas” package to operate Excel, such as reading, writing, modifying, and deleting data. We used “xlrd” and “xlwt” modules to write JSON data into Excel, omitting to install “pandas” package and saving memory. Our research questions, RQ1 and RQ2 can be solved through developed functions. And the study conducted scored data analysis in SPSS 26, in order to explore what model can be built between scoring variables to powerfully predict Chinese college students’ oral English performance in read-aloud task.

Authors collected 1056 pieces of audio data and employed the Youdao ASS cloud server to score by running the five functions in Pycharm. Then, we got six scored parameters (i. e., “pronunciation”, “fluency”, “integrity”, “speed”, “duration”, and “overall”). Authors randomly chose 20 pieces original audio files to ask two raters to give an overall evaluation for every audio and the rater-ASS consistency is higher (r= 0.83), which indicates our Python project runs well. For RQ3, authors analyzed the data in Excel which are partly shown in Figure 3 and found that the test takers’ spoken English performance in read aloud task is mainly influenced by the pronunciation accuracy and word/sentence integrity. This finding is consistent with Li (2008) [30] and Gong (2009) [31] whose findings are ASS can well reflect Chinese students’ weakness on pronunciation accuracy rate, standard degree, and word integrity. Although the fluency is a main metric for EFL learner oral performance, it seems that the fluency little influences students’ oral English performance in our study. It is not in line with Chen, et al. (2018) [12] and Bamdev, et al. [8]. In Chen study, SpeechRater 5.0 scores speech speed and speech flow (both refer to fluency metric) with precise algorithms, and Bamdev [8] also affirmed that fluency is a main domain of hand-crafted linguistic features when studying the relation between the linguistic cues and the participants’ oral English proficiency level. In fact, when Pearson correlation analysis was conducted in the study, the strong correlation exists between “pronunciation” and “fluency” (r = 0.885). After conducting regression analysis, we found that the linear relationship also exists between “pronunciation” and “fluency” (R = .885, R2 = .783; F (1, 963) = 3478.499, p = .000), which indicates that 78.3% changes of “fluency” is affected by “pronunciation”. It seems that “fluency” can be replaced by “pronunciation” and cannot be a main factor to predictor teat takers’ oral performance in our study and in our study.

The reasons of these inconsistencies are that 1) our audio data only refers to read-aloud task and there are few pauses and silence (both are features of fluency) in students’ audio files; and 2) the words in read- aloud are so common that students don’t need to take time to think how to pronounce. Therefore, the total score is little influenced by fluency. Our results imply that the difficulty of the test task can affect the fluency of EFL learners’ oral performance.

Out study can help teachers to learn about their students’ spoken English proficiency in time by running the five functions. Teachers could modify their teaching methods and course contents by analyzing the overall proficiency of students’ oral English. Moreover, our study would be also helpful for Chinese as a second language (CSL) learners to test or practice in and out of class and get a scored result in time.

Nowadays, ASS cloud services are a powerful auxiliary tool for oral evaluation, but this technology will still show some differences from manual scoring in open-question- answer tests (Jiang &Chen, 2021), so ASS can help students to improve their oral pronunciation and vocabulary, but its function in sentence logic composition and semantic analysis needs to be improved, and it has certain limitations to score high-level oral proficiency (Luo &Han, 2014). The functions compiled in this study can be used for referenced by teachers in large number of oral English test for helping them analyze the test results in their studies. In addition, the data analysis of this paper summarizes the most powerful predictors of college students 'oral performance in a certain test task, which provides a reference for oral teaching.

There are still some limitations in our study. Firstly, the codes of this study need to be optimized to improve the performance of the program. Moreover, more functions can be developed to collect the returned feedbacks from the ASS cloud server, and convert them into easily readable

ISSN: 2710-0170

texts or data for further study. Secondly, data collection is only selected from read aloud task (one of the many test tasks in the oral test), and does not involve other tasks. The next research will collect data from other test tasks for comprehensive analyses, in order to summarize the characteristics of oral English of college students and their advantages and disadvantages in the various tasks of the oral test, and this will provide more resources for teaching and research in the future.

Acknowledgments

Authors would like to acknowledge co-workers for collecting the audio data. Authors are grateful to some teachers for their help to rate some audio data. Authors also thank students for their contributions to the speech data.

Funding Statement: The study has been approved by Education and Teaching Project (No. XGH21053) from Shaanxi Higher Education Society, by the by the Experimental Technique project (No. SY20220217) from Northwest A&F University, and by the Division of Higher Education Industry-University Cooperation Collaborative Educational Programs-- "Research on the Reform of Simulation Experimental Teaching of English Speaking in an Immersive Style" (220501867114542) and "Research on the Construction of English Translation Training Platform and Corpus" (2205050867114542) by the Ministry of Education.

References

[1] Cucchiarini, C, Strik, H, and Boves, L (2000b). Quantitative assessment of second language learners' fluency by means of automatic speech recognition technology. Journal of the Acoustical Society of America, 107(2) 989–999. DOI:10.1121/1.428279.

[2] Derwing, TM, Munro, MJ, and Carbonaro, M (2000). Does popular speech recognition software work with ESL speech? TESOL Quarterly, 34(3) 592–603. DOI:10.2307/3587748.

[3] Bernstein, J, Van Moere, A and Cheng, J (2010). Validating Automated Speaking Tests. Language Testing, 27(3) 355–377. DOI:10.1177/0265532210364404.

[4] Evanini, K, Heilman, M, Wang, X and Blanchard, D (2015). Automated scoring for the TOEFL Junior comprehensive writing and speaking test. ETS Research Report Series, (1) 1–11. DOI:10.1002/ets2.12052.

[5] Hayashi,Y, Kondo, K and Ishii, Y (2023) Automated speech scoring of dialogue response by Japanese learners of English as a foreign language, Innovation in Language Learning and Teaching. DOI: 10.1080/17501229.2023.2217181.

[6] Hirai, A, Kondo, Y .and Fujita, R. (2021). "Development of an Automated Speech Scoring System: A Comparison with Human Raters." Language Education & Technology, 58 17–41. DOI:10.24539/let.58.0_17.

[7] Zechner, K and Loukina, A. (2020). "Automated Scoring of Extended Spontaneous Speech." Handbook of Automated Scoring: Theory into Practice. Yan, D, Rupp, AA and Foltz, PW. London, Chapman and Hall/CRC. [Crossref].

[8] Bamdev, P, Grover, MS, Singla, YK, and Vafaee, P, et al. (2023). Automated Speech Scoring System Under The Lens. International Journal of Artificial Intelligence in Education, 33 119–154. DOI:10.1007/s40593-022-00291-5.

[9] Li, X, Li XM, Chen SH, Ma SH and Xie, FF (2022) Neural-based automatic scoring model for Chinese-English interpretation with a multi-indicator assessment. Connection Science, 34(1) 1638-1653. DOI: 10.1080/09540091.2022.2078279.

[10] Ockey, GJ, Gu, L and Keehner, M (2017). Web-based virtual environments for facilitating assessment of L2 oral communication ability. Language Assessment Quarterly, 14(4), 346–359. DOI:10.1080/15434303.2017.1400036.

ISSN: 2710-0170

[11] Downey, R, Farhady, H, Present-Thomas, R, and Suzuki, M, et al. (2008). Evaluation of the usefulness of the Versant for English test: A response. Language Assessment Quarterly, 5(2) 160–167. DOI:10.1080/15434300801934744.

[12] Chen, L, Zechner, K, Yoon, S, and Evanini, Y K, et al. (2018). Automated scoring of nonnative speech using the speechrater sm v. 5.0 engine. ETS Research Report Series, (1): 1–31. DOI:10.1002/ ets2. 12198.

[13] Loukina, A, Zechner, K, Chen, L, and Heilman, M (2015). Feature selection for automated speech scoring. 10th Workshop on Innovative Use of NLP for Building Educational Applications. Stroudsburg, PA, Association for Computational Linguistics. [Crossref].

[14] Yoon, S and Zechner, K (2017). Combining human and automated scores for the improved assessment of non-native speech. Speech Communication, 93: 43-52. DOI: 10.1016/j.specom.2017. 08.001.

[15] Bernstein, J, Cohen, M, Murveit, H, Rtischev, D, et al. (1990) Automatic evaluation and training in English pronunciation. 1990 International Conference on Spoken Language Processing, Kobe, Japan, ProcICSLP-90. [Crossref].

[16] Bernstein, J, Van Moere, A. and Cheng, J (2010). Validating Automated Speaking Tests. Language Testing, 27 (3) 355–377. DOI: 10.1177/0265532210364404.

[17] Eskenazi, M (2009). An overview of spoken language technology for education. Speech Communication, 51(10) 834-844. DOI: 10.1016/j.specom.2009.04.005.

[18] Evanini, K., Cogan, H. and Hakuta, K. (2017). Approaches to Automated Scoring of Speaking for k-12 English language proficiency Assessments. ETS Research Report Series, (1):1-11. DOI: 10.1002/ets 2.12147.

[19] LeCun, Y, Bengio, Y and Hinton, G. (2015) Deep learning. Nature, 521 (7553) 436–444. DOI:10.10 38/nature14539.

[20] Li, H (2018) Deep learning for natural language processing: advantages and challenges, National Science Review, 5 (1) 24–26. DOI: 10.1093/nsr/nwx110.

[21] Zhao, W, Liu, X, Jing J and Xi, R (2022). Re-LSTM: A long short-term memory network text similarity algorithm based on weighted word embedding. Connection Science 34 (1) 2652-2670. DOI:10.10 80/09540091.2022.2140122.

[22] Witt, SM and Yong S.J (2000) Phone-level pronunciation scoring and assessment for interactive language learning. Speech Communication, 30(2–3) 95-108. DOI:10.1016/S0167-6393(99)00044-8.

[23] Zechner, K, Higgins, D, Xi, X and Williamson, DM. (2009). Automatic Scoring of non-Native Spontaneous Speech in Tests of Spoken English. Speech Communication, 51 (10): 883–895. DOI:10.1016/j.specom.

[24] Pearson Education (2011). Versant TM English Test: Test Description & Validation Summary; Palo Alto; CA; Pearson Knowledge Technologies. DOI:10.1016/S0167-6393(99)00044-8.

[25] Xi, X, Higgins, D, Zechner, K and Williamson, D (2012). A comparison of two scoring methods for an automated speech scoring system. Language Testing, 29(3) 371–394. DOI: 10.1177/02655322114 25673.

[26] Jiang, JL and Chen D (2021). Study on ASS of Subjective Questions: Review, Reflection and Prospect J. Chinese Foreign Language, (06) 58-64. [Crossref].

[27] Lou, K and Han, B (2014) Review and enlightenment of Ordinate and SpeechRater ASS system. Technology Enhanced Foreign Language Education, (04) 27-32. [Crossref].

[28] Bernstein, J, Cheng, J (2007). Logic and validation of a fully automatic spoken English test. The Path of Speech Technologies in Computer Assisted Language Learning: From Research Toward Practice. Holland, V.M. and Fisher FP, Florence, Routledge. [Crossref].

[29] Sun, H (2021). A review of auto-scoring of spoken English at home and abroad. Foreign Language Education in China, (02): 28-36 + 89-90. [Crossref].

ISSN: 2710-0170

[30] Li, M, Yang, X, Feng, G and Wu, M, et al. (2008). Feasibility study and practice of large-scale college English oral test reading machine. Foreign Language world, (04) 88-95. [Crossref].

[31] Gong, L, Liang, W, and Ding Y (2009). Feasibility analysis and Practice research of using machine marking for large-scale Oral English test and reading questions. Technology Enhanced Foreign Language Education, (2) 10-15. [Crossref].

[32] Lv, M (2015), The exploration and practice of Intelligent assessment technology in large-scale oral English test marking. Chinese Examination, (10) 51-57. DOI:10.19360/j.cnki.11-3303/g4.2015.10.009.[Crossref].