Public Speaking Web Trainer

Daniil Pliushchenko
Higher School of Economics
Saint Petersburg, Russia
plyushenko@bk.ru

Mark Zaslavskiy
Saint Petersburg Electrotechnical University "LETI",
JetBrains Research
Saint Petersburg, Russia
mark.zaslavskiy@gmail.com

Abstract—This paper describes a web-based application that can be used to prepare for giving a public talk. Firstly, an overview of existing public speaking applications is provided. We noticed that existing applications lacked evaluation customization, presentation file handling features, and integration with external services such as learning management systems. This application provides feedback based on a pack of criteria such as speech duration, speech pace, filler words usage, and others. We propose flexible application architecture so new steps of evaluation can be added, removed, or changed, also it can be scaled to support potentially increasing application load. We measured training processing time both for a single training and a set of trainings submitted to the application simultaneously. So far the longest step of processing is speech recognition so configurations with different number of speech recognition instances were considered to get faster training processing time.

I. Introduction

A large number of students and researchers give public talks with their presentations, for instance, a thesis defense or a speech at a conference. Usually, they do some training before the actual talk. In order to evaluate the quality of a speech and a presentation, there must be at least one other person (besides the actual speaker themself) who should have a high level of expertise in a particular domain. Moreover, this is a time-consuming action, although some preliminary checks can be done automatically.

The goal of this work is to design and develop an interoperable, scalable in terms of potential load and possible steps of evaluation, open-source web-based application that does the preliminary evaluation of a given speech with a presentation. The usage scenarios include both individual training and handling cases when each student from a group has to complete several attempts and gain the required number of points as a part of preparation for their thesis defense, mostly giving presentations and talks in Russian.

This paper is structured as follows. In section II related works such as existing applications that can improve public speaking skills and speech recognition libraries are reviewed. Next, in Section III the overall structure of the application is described. The measurement results and the conclusion with further plans are presented in sections IV and V, respectively.

II. Related work

A. Existing applications

In order to discover analogs, applications that somehow evaluate speaking and/or pronunciation skills were considered. According to the goal of this work, ability to integrate with external tools, such as having an API or interoperability with LMS (Learning Management System), LTI (Learning Tools Interoperability) [1] support, as well as flexible configuration, ability to attach a presentation, and speech recording were criteria for the comparison. Keywords and key phrases such as 'speech', 'pronunciation', 'speaking', 'evaluation', 'analysis', 'public speaking training', 'public speaking application', 'presentation' were used for search.

Kopf et al. [2] describe application for presentation skills evaluation with the main focus on speaker's behavior rather than on presentation and speech content. Criteria such as hand and arm gestures, eye contact with the audience, speaker's pose, and movement are considered by using Microsoft Kinect as an input sensor. The authors also note that speech pace and talk per slide duration are important, but they do not process the actual content or provide any feedback based on it.

Hanani et al. [3] present a framework that extracts features from multiple factors such as speech, presentation, and body language. After the extraction, several machine learning methods for classification are applied. Each feature value is classified as a high or low in one part of the experiments and as a high, average or low in the other part, but it is still impossible to choose a desired criteria subset and adjust the criteria.

Speechace API [4] provides evaluation for American English and British English speech. The feedback is provided for each sound and each syllable whether it was similar to a reference pronunciation or, if not, why. Also, word count, syllable per second, pause statistics, correct word ratio, and other statistics can be retrieved via API. Speechace is the only discovered tool having LTI integration.

Voice notebook [5] is a web-based set of speech recognition tools. It also has mobile apps converting speech to text with timestamps. One of the tools is the reading and pronunciation test where the user provides a text and then is reading it sentence by sentence while their voice is being recorded. Evaluation represents the percentage of recognized words and completeness of recognized words for the last

spoken sentence and the whole text. Another tool provides a chosen number of versions of recognized texts with their likelihoods.

Speakit [6] is an Android application that allows you to check your American English pronunciation. Words and phrases can be looked up in the dictionary or found by sound type, (i.e., short vowels, long vowels, diphthongs, etc.) or a category (i.e., food, family, friends, etc.). It is possible to record an attempt at the pronunciation of that word or phrase and receive feedback saying whether it was similar to a reference pronunciation or not.

Aksent [7] is an iOS application that calculates the similarity percentage between a reference pronunciation and a user pronunciation in one of more than twenty supported languages. A word or a phrase can be typed and translated to a target language.

Speeko [8] is a paid iOS application. It analyses a speech in terms of pace, eloquence, pausing, intonation, and articulation providing metrics such as the number of words per minute and frequency statistics for each spoken filler. It is also possible to obtain a transcription and a record of the speech. A user can set a duration limit and jot down their notes that will be shown during the speech record.

LikeSo [9] is a paid iOS application that helps to eliminate fillers from your speech. A user can set a record duration (up to 30 minutes) and a subset of fillers that would affect the score. The total score is a ratio of non-fillers in the speech, also frequency statistics of spoken fillers, number of total words spoken, and words per minute speed are provided in the feedback.

ELSA Speak [10] is a mobile application available both for Android and iOS helping to speak American English. The application contains several topics with phrases and words related to it. User record is checked sound by sound whether it is similar to the reference pronunciation or not, also the total score representing the whole word/phrase similarity is provided.

Orai [11] is a paid mobile application available both for Android and iOS. Orai provides recorded speech analysis including pace statistics, filler statistics, and conciseness issues (such as repeated words and so on). It is also possible to store speech notes and select notes to be shown during speech recording.

Говорилло (Govorillo) [12] is an Android app that allows user to record their speech and receive feedback that contains pace, fillers ratio, and speech complexity. The latter metric returns the lower bound of the listener's age who would be comfortable to understand the speech.

Even though existing applications provide speech evaluation, they lack the ability of presentation attachment, criteria customization, and external tools integration. That makes it hardly possible to use them for the above-mentioned usage scenarios. Existing applications overview is shown in Table I.

B. Speech recognition libraries review

Since presentation recordings should be processed to extract information from the user's speech, speech recognition libraries are compared in this section. The comparison is based on offline access, Russian language processing support, provision of timestamps, and price. The optimal speech recognition system would be available offline in order to get rid of external dependency, support the Russian language because this will be the main presentation language, provide timestamps for each spoken word to match a speech to the presentation slides and to calculate statistics, and be free because the entire project is intended to be open-sourced and free. Libraries overview is shown in Table II.

The libraries that match the above-mentioned filters are Vosk [13] and Speech-To-Text (Russian) [14]. Those two libraries along with Wit.ai [15] were considered in a more detailed way by comparing the quality of speech recognition using the w-shingle algorithm [16]. Shingles are overlapped sentences of the text, consisting of a fixed length of words. Recognition quality was measured as an average ratio of correctly recognized shingles in the dataset. The dataset was collected to match this work's goal and consists of ten presentations given in the Russian language by five male and five female speakers during their thesis defense or online lessons. Vosk library was chosen because it has the best voice recognition results, as shown in Table III.

III. Solution

A. Project structure

Since existing tools are hardly applicable to handle the usage scenarios, we propose the project structure described below.

The project consists of several services. The user-facing one is a Flask web application where the user can upload their presentation, set up the speech recording, and start training. After the training, the speech recording and the presentation processing happen, then the user gets the training results, statistics, and the overall feedback. Since modules with a variety of execution time and latency might be added to the system as part of training processing, an asynchronous approach is used so other services are queue-based, they are dedicated to:

- Speech recordings recognition. Identifiers of speech recordings that should be recognized are sent to that service. Workers extract identifiers from the queue and send recordings to a speech recognition system. It returns a file containing information about words that are sent to a file storage. The identifier of that file is sent to a queue dedicated to recognized speech processing (described below).
- Presentations recognition (parsing). It works the similar way as the previous service, but presentation files are recognized using PyMuPDF [23] library.
- Recognized speech processing. Identifiers of files with information about recognized speech are sent to that service. Workers extract identifiers from the queue and call the file processing, such as slide split and statistics calculation (both per-slide and for the entire file).

TABLE I. EXISTING APPLICATIONS OVERVIEW

Name	Platforms	API?	Presentation attachment	Speech duration limits	Speech recording	Languages	Custom criteria?	Paid?
Speechace	Web	Yes	No	15 seconds for free	Yes	English	No	Free demo
Voice Notebook	Web	No	No	No	Yes	8	No	No
Speakit	Android	No	No	One phrase	Fixed words	American English	No	No
Aksent	iOS	No	No	One phrase	Yes	20+	No	No
Speeko	iOS	No	Text notes for the current record	No	Yes	English	No	Free trial
LikeSo	iOS	No	No	30 minutes	Yes	English	Subset of fillers	Yes
Orai	iOS, Android	No	Text notes	No	Yes	English	Subset of fillers	Yes
ELSASpeak	iOS, Android	No	No	One phrase	Fixed words	American English	No	Free demo
Говорилло (Govorillo)	Android	No	No	No	Yes	Russian	No	Yes
Public Speaking Web Trainer	Web	Yes	Yes	No	Yes	Russian	Yes	No

TABLE II. VOICE RECOGNITION LIBRARIES OVERVIEW

Name	Offline?	Russian language?	Timestamps?	Price
Vosk	Yes	Yes	Yes	Free
Speech-to-Text (Russian)	Yes	Yes	No	Free
Picovoice [17]	Yes	No	No	Free
At16k [18]	Yes	No	Yes	Free for personal use
Google Web Speech API [19]	No	Yes	Yes	Free one hour per month
Google Cloud Speech API [20]	No	Yes	Yes	Free one hour per month
Microsoft Speech Service [21]	No	Yes	Yes	Free 5000 requests per month
IBM Speech to Text [22]	No	No	Yes	Free 50 hours per month
Wit.ai	No	Yes	Yes	Free but query length limit is 20 seconds, no more than 60 requests per minute

TABLE III. VOICE RECOGNITION LIBRARIES COMPARISON

Shingle size	1		2		3		4	
	Average, %	Standard deviation, %	Average, %	Standard deviation, %	Average, %	Standard deviation, %	Average, %	Standard deviation, %
Vosk	74.84	10.03	60.27	12.28	50.08	12.94	42.24	12.80
Speech-to-Text (Russian)	49.14	14.41	28.27	12.33	17.78	10.34	11.82	8.30
Wit.ai	55.29	11.70	35.70	10.99	25.47	9.76	18.85	8.57

- Recognized presentation processing. It works the similar way as the previous service but files with information about recognized presentations are processed.
- Trainings processing. A set of criteria is applied, then feedback is calculated based on the criteria results.
- Passing training results back to an LMS via LTI when feedback is ready.

Those services run in separate docker containers that are controlled via docker-compose. The application architecture is shown in Fig. 1. Each rounded rectangle represents a separate docker container, modules represented as gray rectangles do not belong to the application.

When the user finishes their training, the presentation file and the presentation record file are sent to a file storage, then their identifiers are sent to the recognition services where they are added to the queues for transformation to the intermediate ‘recognized’ state and to the ‘processed’ state, then the training can be checked against a set of criteria.

Each criterion returns a float number depending on whether the training fits that criterion or not. Also, each criterion contains information about dependent criteria

because some criteria results may be affected by them. At the moment, available criteria check that speech does not last too long, and speech pace matches boundaries by comparing values that were retrieved by recognized speech processor service. Another criterion checks that speech does not contain fillers. Currently, the criterion detects whether transcription contains elements from a predefined list filled with well-known words and phrases that tend to be considered as fillers. There is also a criterion for checking the absence of speech recordings, similar to those previously recorded.

It is possible that a particular criterion should be applied in a slightly different way. That is why the ‘parameterized criterion’ entity is introduced. It contains the exact parameters to calculate a criterion. For instance, parameterized criterion checks whether the speech duration exceeds 7 minutes or not (that can be changed to any required duration, if needed). Since training should be checked against an arbitrary set of criteria, parameterized criteria are combined into criteria packs containing information about the order of applied criteria. When all criteria results are present, a feedback evaluator is used to calculate the overall training grade. Feedback evaluator contains weights of each criterion to be able to treat results of the same criteria pack in a different way, for example, to make a criterion the blocking one or to emphasize the importance of a particular criterion.

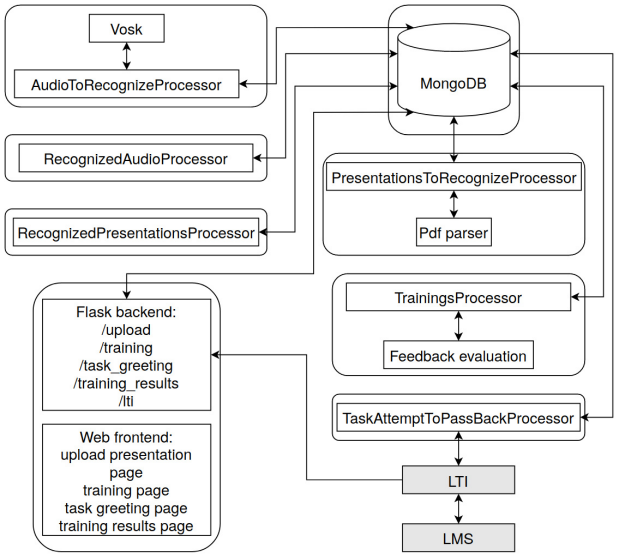

Fig. 1. Application architecture

B. LTI support

LTI protocol is supported by many popular learning management systems and massive open online courses platforms such as Moodle, Blackboard Learn, Stepik.org, edX.org, etc. and can be used to exchange task information and feedback between learning tools, as well as perform authorization. Several steps should be done to integrate the application with a learning management system via LTI. Firstly, a secret key should be shared between two systems. Secondly, a task should be created in the LMS. The entry point is a POST HTTP request coming from a learning management system to a dedicated application endpoint. That request contains information about the task, the user, and authorization data including the shared secret key. Also, the request contains a link for passing the task grade back to the LMS. If the request is valid, an application session record is created, otherwise, the request is rejected.

An example of a task can be a preparation for a thesis defense. A student should complete a given number of training attempts and gain the required number of points. After doing so, the overall grade is calculated and passed back to the LMS. Data pipeline overview is shown in Fig. 2.

C. Data model

MongoDB [24] database along with GridFS file storage is used because the further structure of data might change due to new ways of training processing. NoSQL databases, such as MongoDB, allow handling that easier because they do not require predefined table structure and relations between them.

The main collection is ‘Trainings’. Username, presentation file and presentation record file (in ‘raw’, ‘recognized’, and ‘processed’ states) identifiers, processing status (of presentation, audio, and the entire training) and their last update timestamps, slide switch timestamps, criteria pack identifiers, task attempt identifiers, feedback evaluator identifiers, and feedback entries are stored there.

‘PresentationFiles’ collection stores presentation fileidentifiers, filenames, and preview file identifiers. The presentation file preview is shown on the page with available presentations.

‘Consumers’, ‘Sessions’, and ‘Tasks` collections are used to handle data related to different LTI consumers (learning management systems, for instance), user sessions, and tasks (such as identifiers, descriptions, and number of attempts), respectively.

‘TaskAttempts’ collection contains information about a username, a task identifier, a set of trainings, such as their identifiers and scores, that were taken to complete a task, and a set of parameters that are used to pass a grade back. Other collections are used to pass a grade back. Other collections are used as queues described above.

IV. Evaluation

To understand how many resources are consumed during training processing, especially in the case when multiple trainings are processed simultaneously, 8 presentations and 8 speech recordings from bachelor, master, and Ph.D. students from FMCS of SPb HSE, and MO EVM, FKTI of SPb ETU were taken. Those materials were used for their thesis defense, course project defense, or preparations for them.

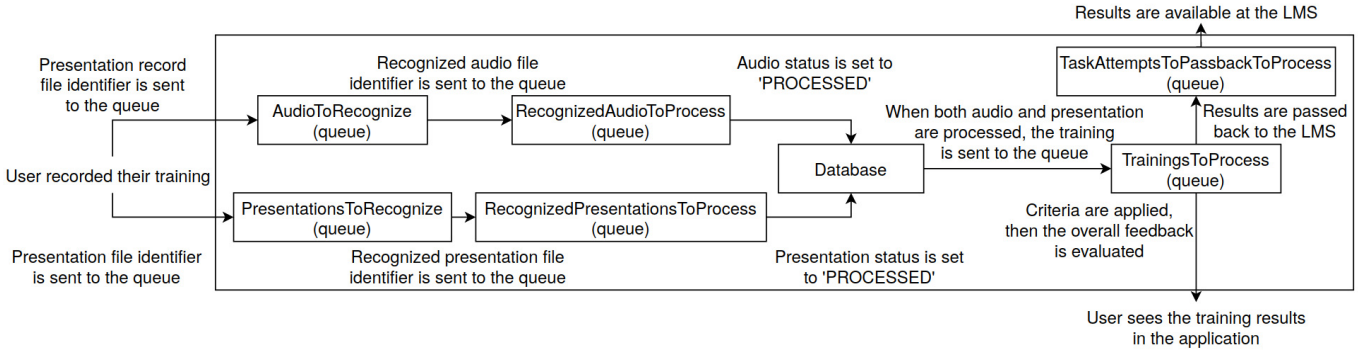

Fig. 2. Data pipeline overview

The following setup was used:

- Lenovo Ideapad L340-15API 81LW005BRU with 12GB RAM and AMD 3200U 2.6 GHz CPU.
- Ubuntu 18.04.

So far the longest step is audio recognition, presentation parsing takes a couple of seconds, and criteria evaluation takes less than a second, hence the overall training processing time is almost equal to the audio processing time. To measure execution time, we used a script written in Python that uploads trainings with the corresponding presentation and speech recording files, then calculates the difference between the timestamp at the beginning of processing and at the end. Measurement results for speech recognition and presentation parsing of components of a single training are shown in Table IV and Table V, respectively. For each sample, processing time does not exceed half of the speech duration.

RAM consumption is estimated as 2.1GB per each Vosk speech recognition instance and a queue associated with it plus around 300MB for other services described in Section III.A.

To evaluate how long it would take to process multiple trainings, 12 copies of the same training that lasts 7 minutes and 39 seconds were taken. Since the longest part of processing is audio recognition, configurations with different number of speech recognition instances were considered.

The measurement results are shown in Table VI. The more speech recognition instances are present, the faster processing goes, and the bigger training part is processed per second of the application's operation.

TABLE IV. SPEECH RECOGNITION DURATION

Speech duration, seconds	Speech recognition duration, seconds	Standard deviation, seconds
436	139.74	0.42
459	144.09	0.49
492	155.40	0.63
541	154.87	0.51
568	168.77	0.68
614	226.37	0.82
810	232.43	3.14
1151	324.33	5.55

TABLE V. PRESENTATION PARSING DURATION

Number of slides	Presentation parsing duration, seconds	Standard deviation, seconds
9	1.29	0.12
10	1.52	0.03
10	1.55	0.09
11	1.50	0.12
15	1.97	0.12
17	2.48	0.12
18	2.59	0.10
20	2.73	0.04

TABLE VI. TRAINING PROCESSING DURATION

Number of speech recognition instances	Total processing time, mm:ss	Training time processed for 1 second, seconds
1	25:31	3.60
2	17:47	5.16
3	14:31	6.32

V. CONCLUSION

In this paper, we describe a web-based application that is dedicated to public speaking preliminary evaluation. The application can be used for training without involving another person and obtaining score based on a given criteria set. The architecture and data pipeline description are followed by the existing applications intended to improve the public speaking skills overview. The project is open-sourced and available at [25]. Currently, several simple criteria are implemented, so we are going to add more criteria, however the application architecture allows adding new criteria easily. Moreover, we plan to conduct usability studies, collect user feedback and analyze it in order to study the effectiveness of the application.

REFERENCES

[1] Learning Tools Interoperability Core Specification 1.3 [Online]. Available: https://www.imsglobal.org/spec/lti/v1p3 [Accessed: 07.03.2021]

[2] Kopf, S., Schön, D., Guthier, B., Rietsche, R. & Effelsberg, W. (2015). A Real-time Feedback System for Presentation Skills.

[3] Hanani, A., Al-Amleh, M., Bazbus, W. & Salameh, S. (2017). Automatic Estimation of Presentation Skills Using Speech, Slides and Gestures.
[4] SpeechAce API [Online]. Available: https://www.speechace.com/#api [Accessed: 21.10.2020].
[5] Voice Notebook [Online]. Available: https://voicenotebook.com/ [Accessed: 21.10.2020].
[6] Speakit [Online]. Available: https://play.google.com/store/apps/details?id=com.eapp.pc [Accessed: 21.10.2020].
[7] Aksent [Online]. Available: https://aksent.ai/ [Accessed: 21.10.2020].
[8] Speeko [Online]. Available: https://www.speeko.co/ [Accessed: 21.10.2020].
[9] LikeSo [Online]. Available: https://apps.apple.com/us/app/likeso/id1074943747 [Accessed: 21.10.2020].
[10] ELSA Speak [Online]. Available: https://play.google.com/store/apps/details?id=us.nobarriers.elsa [Accessed: 21.10.2020].
[11] Orai [Online]. Available: https://apps.apple.com/us/app/orai-improve-public-speaking/id1203178170 [Accessed: 21.10.2020].
[12] Говорилло (Govorillo) [Online]. Available: https://play.google.com/store /apps/details?id=com.vsquad.projects.govorillo [Accessed: 21.10.2020].
[13] Vosk [Online]. Available: https://alphacephei.com/vosk/ [Accessed: 13.10.2020].
[14] Speech-To-Text (Russian) [Online]. Available: https://github.com/SergeyShk/Speech-to-Text-Russian [Accessed: 13.10.2020].
[15] Wit.ai [Online]. Available: https://wit.ai/ [Accessed: 13.10.2020].
[16] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press.
[17] Picovoice [Online]. Available: https://picovoice.ai/ [Accessed: 13.10.2020].
[18] At16k [Online]. Available: https://at16k.com/ [Accessed: 13.10.2020].
[19] Google Web Speech API [Online]. Available: https://wicg.github.io/speech-api/ [Accessed: 13.10.2020].
[20] Google Cloud Speech API [Online]. Available: https://cloud.google.com/speech-to-text [Accessed: 13.10.2020].
[21] Microsoft Speech Service [Online]. Available: https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/ [Accessed: 10.03.2021].
[22] IBM Speech to Text [Online]. Available: https://www.ibm.com/cloud/watson-speech-to-text [Accessed: 13.10.2020].
[23] PyMuPDF documentation [Online]. Available: https://pymupdf.readthedocs.io/en/latest/ [Accessed: 10.03.2021].
[24] Banker, K. (2011). *MongoDB in action.* Manning Publications Co.
[25] OSLL/web_speech_trainer [Online]. Available: https://github.com/OSLL/web_speech_trainer [Accessed 10.03.2021]