Tonal Star: Learning tonal languages through speech visualisation

Tianyi Chen Sam Zandbergen Janne Spork Xander Verstraeten
Department of Industrial Design, Eindhoven University of Technology, The Netherlands
{t.chen@student.tue.nl, s.l.zandbergen@student.tue.nl, x.verstraeten@student.tue.nl, m.j.spork@student.tue.nl}

ABSTRACT
For non-native tonal language speakers, it is very difficult to pronounce and comprehend the correct pronunciation of the tones. This complication is very important in the sense that if the tones are not pronounced accordingly, the language cannot be spoken correctly and big misunderstandings can occur. To mitigate this complication, a combination of speech visualization through the means of a frequency spectrogram and text to speech algorithm are used. Resulting in the study software called "Tonal star" enabling the user to practice their pronunciation, by reference checking their pronunciation with that of a native speaker. Based on the above findings the research shows insight into the use of visual feedback for the pronunciation of tonal languages.

Author Keywords
Machine learning; live feedback; tonal language; neural network; voice recognition; explainable AI.

INTRODUCTION
The learning of tonal languages
From all the languages in the world, the category of tonal languages is seen as one of the most difficult. Languages like Chinese, Vietnamese and Japanese fall under this category, containing different pitches to correctly pronounce the language. These tones are especially hard to learn for non-native tonal language speakers. (Hao, 2012)
Therefore the following research question is formed: To which degree does Explainable Artificial Intelligence help to understand the pronunciation of tonal languages through visual feedback?
To answer this research question a new program called "Tonal star" will be developed, that gives visual feedback on the pronunciation of the user. Hereby the user can reference check if they correctly pronounce the tones compared to native pronunciation. This visual feedback will be supported by speech to text, making it possible to give a percentage of how correct the pronunciation of the user is.
What we as the researchers expect is that giving visual feedback will simplify the learning of tonal languages. Filling the gap of not knowing the details of your performance; which is, with the currently available tools, based on the user's own interpretation. The impact of Tonal Star can make the learning of tonal languages more accessible for all the non-native tonal speakers in the world which accumulates up to almost 60% for the world population (Matthew, z.d.). With the use of Explainable artificial intelligence (XAI) the system provides insights into the data, making the user aware of the decisions made.

Related work
To get a better grasp on the aspect of tonal languages, we take Chinese as the language reference point within this research paper. The Chinese language contains 4 spoken tones as illustrated in Figure 1.

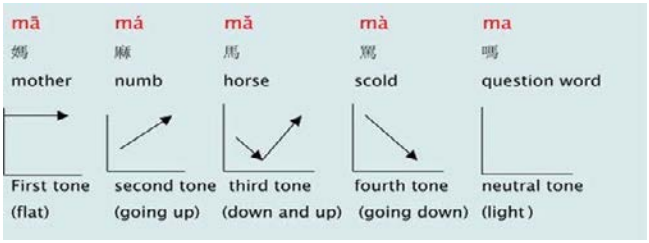


Figure 1: Illustration of the 4 tones in Chinese (Tones in Mandarin Chinese, 2020)

Every spoken word in a sentence has to be pronounced with its given tone, to correctly construct the intended sentence. Figure 1 also illustrates the importance of tones, as a small difference in pronunciation can change a word's meaning, just like with the word "ma".

Translation software like "Google translate" uses an artificial neural network combined with other algorithms, to convert speech to text (Wu, 2016). This technique allows users to directly translate spoken Chinese to a desired other language. To incorporate tones within the neural network for translation, the system has been trained with a lot of diverse data from native tonal speakers. To train the system on these tones the voices of people have been transformed from analog speaking to a digital spectrogram for analyzing. Figure 2 shows the pitch contour of the second tone in Chinese build up from 10 segments. These segments are connected to six hidden neurons. Where afterwards the neurons are followed up by transfer functions, indicating the corresponding 1th, 2th, 3th and 4th chinese tone (Li et al., 2006, p. 31).

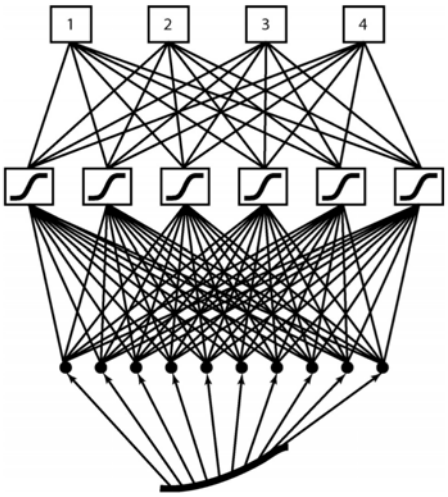


Figure 2: Pitch contour of the second tone in Chinese build up from 10 segments (Li et al., 2006, p. 31)

To visualise the frequency of the different Chinese tones, a spectrogram can be made through multiple fast fourier transform formulas on the analog speech (Kan & Ito, 2020, p. 42). Figure 3 shows how this visualisation can be created for the chinese word "shu" with its different tonal pronunciations.

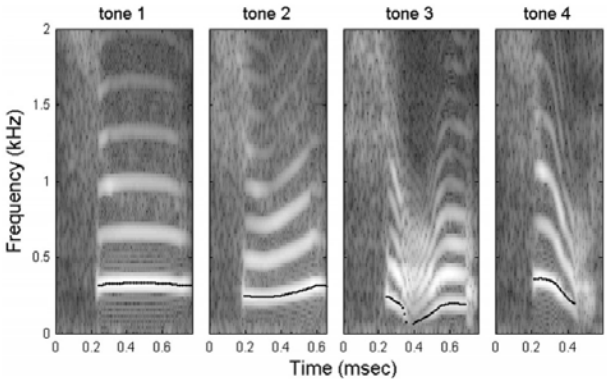


Figure 3: Visualization of the tones (Li et al., 2006, p. 31)

There are different programs like Duolingo and Ponddy to learn tonal languages with. They use machine learning for keeping track of what the user has learned so far and what needs more practise (Settles et al., 2020). During pronunciation exercises a comparison is made in the back-end of the software to see if your pronounced sentence or word matches the intended sentence or word. The users themselves do not get any indication of how they pronounced the tones. Currently the only option of feedback is a right or wrong indication, by repetitive mimicking of the given audio fragment in each exercise.

METHODS AND MATERIALS

To overcome the difficulty of learning the tones a combination of visual feedback with speech to text and tone pronunciation has been introduced. Earlier research done on visual feedback, indicates that this kind of feedback works especially well for learning tonal languages (Hincks & Edlund, 2009).

Visual feedback of the pronounced tones can also go under the name of "audiovisual education". There is an advantage in using methods like this for education but there also might be a downside. Training needs to be provided to fully understand the audiovisual aids within Tonalstar (Rasul et al., 2011, p. 79). The initial spectrograms for users to look at as a reference point were pre-recorded by a native Chinese speaker. To reduce generalization error and optimize the system, making the total system work more accurate, a pre-learned algorithm of Google has been implemented (Dahl et al., 2012, p. 40).

The choice of using the pre-learned algorithm of Google over a newly made speech to text algorithm all comes down to accuracy. Google has trained their model extensively over the years with all kinds of different voices as input data, making it a system that can translate with high accuracy. A new algorithm that would need training, will take a lot of time and data that is not available within the time span of the research, to make it as accurate as the algorithm of Google (Aiken, 2019).

To implement the features of Tonal Star, different python functions have been utilized within the main program as shown in appendix I. The main program calculates the percentage of how correct the intendent sentence is pronounced and shows the corresponding spectrogram. The input for the calculation is generated by the function called "recognize_google", from the code "SpeechRecognition" as shown in appendix II. This function utilizes the algorithm of Google to convert what is said into text. Where afterwards it's reference checked to the predetermined sentence.

To generate the spectrogram that affiliates the translation from Google, the function "Record" from the code of "SpeechRecord" shown in appendix III is used. Here the program starts listening to the user and saves what is said in a wav file for later analyzes in the main program. Where the function "plot_speech" from the code in "plotSpectogram" as shown in appendix IV translate forms it into a spectrogram.

Learning algorithm

The learning algorithm that is used within Tonal Star stems from the application programming interface (API) from Google. To make the translation work, a neural machine translation (NMT) system incorporates an end-to-end artificial neural network that performs deep learning. The algorithm learns from millions of examples to improve the quality, also called example-based machine translation (EBMT). The algorithm does not translate word for word but by entire sentences at a time. To get the most proper translation a broader context is used. Where afterwards it rearranges the translation to more of a human speaker where grammar is correct (Li et al., 2006, p. 31).

DESIGN OF TONAL STAR

Design of the interaction

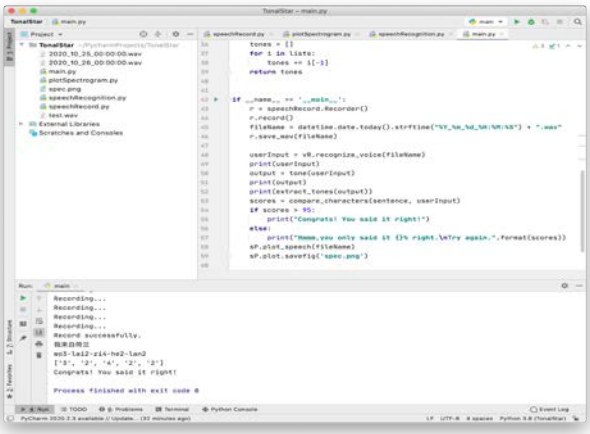
The interaction scenario in Figure 4 shows the combination of the agent's actions and the required interactions with our user group: people who want to learn tonal languages.

Figure 4: User interaction scenario

Tonal Star is an app that uses machine learning to guide a user to learn tonal languages, in this research Chinese. From top to bottom, from left to right: the user opens the Tonal Star application. First, the user uses the app to listen to the perfect tonal pronunciation of a sentence. For example the sentence “我来自荷兰”. The sentence is also shown in text, both in Chinese as the translation of the sentence in the language of the user: "I'm from the Netherlands.". The user presses the record button and tries to pronounce the sentence himself. The Tonal Star app shows two spectrograms to visually compare the perfect pronunciation with the pronunciation of the user. Next to the spectrograms, the percentage in which the user’s pronunciation corresponds to the correct pronunciation is shown. The sentence is also shown in text, both in Chinese as the translation: “我吃三轮车” with the translation “I eat tricycle”. The user is encouraged to try again until a high score is achieved. In the personal overview of Tonal Star, the app gives feedback on the overall performance of the user making use of explainable AI. An overview of the prototype can be found in appendix V.

Intelligent behaviour and embodiment.

The intelligent agent is a tablet where the app is installed. With access to the tablet’s microphone, the app can record audio from the user, which will be encoded and processed by Google speech recognition. In the console of the python code, ‘Recording...’ will be printed until the user stops talking to the microphone, and the ‘Recording successfully.’ triggers a visual feedback on the app indicating that the audio has been successfully saved. Then the speech recognition results gets returned as well as the pinyin of the sentence. These are also shown on the app for users to see what their audio sounds like, and the tones in the form of pinyin. An extra function was provided to extract only the tones as numbers, this is for further visualization for a more vivid display on the user interface. When users speak the sentence correctly, the result of ‘Congrats! You said it right!’ will be printed (Figure 5a.), while on the app the user can see a ‘Perfect’ comment as well as a full bar indicating a good score. When the user does not pronounce correctly, sentences will be compared and a score of correctness will be given(Figure 5b.). For example a 60% percent correct was given when the user said “过来支荷兰” instead of “我来自荷兰”. This score would also be shown on the progress bar as well as a comment of ‘You can do better!’ In either case, the spectrogram will be generated from the audio input and displayed on the app as a visual feedback(Figure 6).



a.

b.

Figure 5: Console running python code, prints a. correct results and b. only partially correct

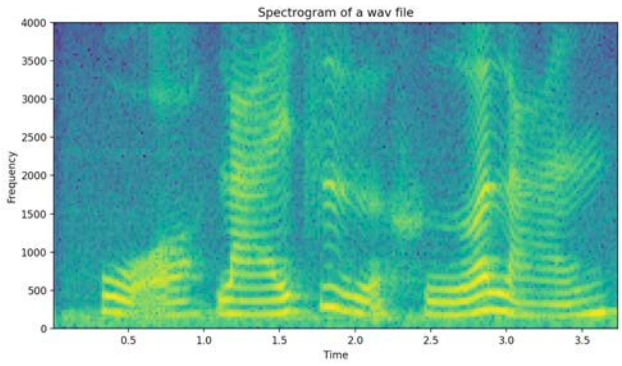


Figure 6: Spectrogram of the sentence "我来自荷兰" (I'm from the Netherlands.)

When users are learning with Tonal Star, it is expected that users cannot pronounce perfectly when they first start with the language. Therefore, it is necessary to understand the user's level and build up a profile, which in time will be improved through machine learning algorithms. Various levels can be provided for the learners to choose their goals and personalized practice with a focus on tones and tone combinations where they make more mistakes. The state diagram of the program is shown in Figure 7.

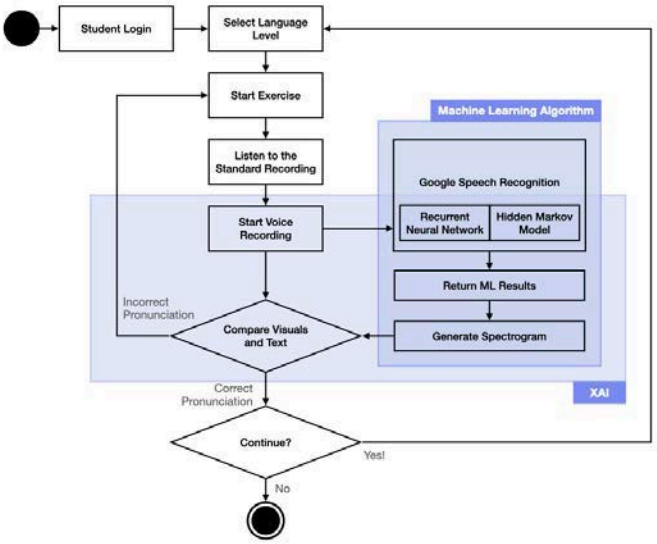
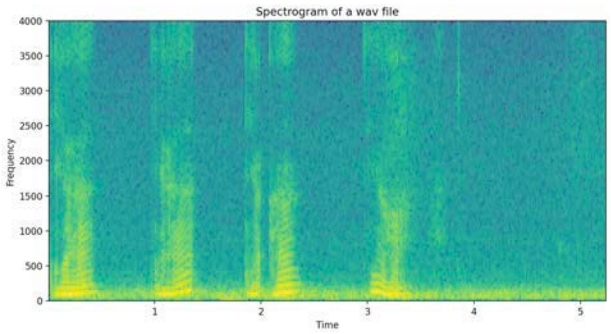


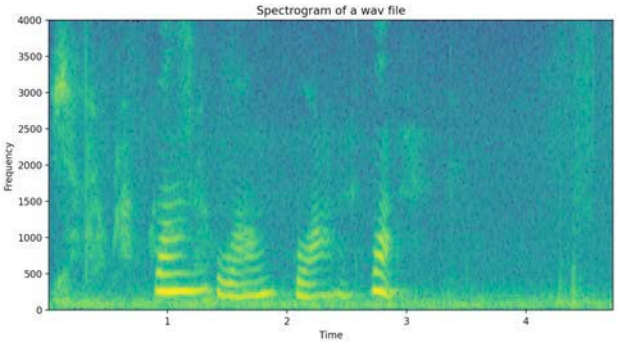
Figure 7: State Diagram of Tonal Star

TESTING AND ANALYSIS

In the initial testing, we compared the spectrograms of different tones between a native speaker and Chinese learner. It can be observed that the spectrogram from a native speaker in Figure 8a. has a clear pattern of the first (high level), second (rising), third (low falling and then rising) and forth (high falling) tones. In Figure 8b. However, when the chinese learner tried to pronounce the tones, the second and third tones are almost identical in terms of spectrograms. In order to validate the results, we played the audio of the chinese learner to the native speaker, asking if she could tell the difference. According to the native speaker, it was rather hard to distinguish the second and third pronunciations when the syllables were played separately, indicating that the insignificant fluctuation in the spectrogram from the chinese learner does in a way reflect the low proficiency of pronunciation.



a.



b.

Figure 8: Spectrograms of the four tone patterns of Mandarin Chinese syllable 'ma'. The speech samples were recorded from (a) female native speaker and (b) male Chinese learner.

Based on this initial testing, we made a testing plan for the intelligent system that includes pairs of native speakers and Chinese learners. We also considered the social context of the smart system where it's mainly for communication between people so the requirements for the precision of pronunciation is less important than the understanding from human brains. Therefore, we decided it would be more convincing when we invite native speakers to the testing for a more humanistic evaluation.

Materials

Sentence clusters in Chinese containing 5-10 short sentences with no longer than 10 characters (in both characters and pinyin). They should be preloaded into the program as the reference sentences.

Participant selection

Participants will preferably be 10 adult native speakers who have lived in China for most of their life without a strong

accent; 10 adult chinese learners who have been learning the language for less than 1 year.

Testing

The participants will first be recorded speaking a testing sentence which will only be used to test the microphone and the working program. During the testing, the native speakers and the chinese learners will be paired up randomly. They will be provided the same sentence clusters and required to speak to each other at medium speed. The native speaker speaks first and then the chinese learner tries to mimic the pronunciation. This will be repeated twice.

Then they will go into different rooms, where they were given a sentence cluster individually and required to speak to the app, where each time the visualized pronunciation is shown on the screen.

Subjective Validation

Each time the native speaker will be asked to grade the chinese learner's pronunciation with a scale from 0 to 5(0 being completely unrecognizable, 5 being native). Then the scores of following only a voice guidance and scores of learning with visualization will be calculated and compared. If the scores have significant differences, it should be possible to conclude that in a more socially relevant context, one learning method is better than the other.

Quantitative Analysis

The spectrograms from the chinese learner will be analyzed with pattern recognition algorithms, to capture the fluctuations of tones. It will then be compared to the spectrogram of the native speaker. If the coselection rates have significant differences, it can provide proof that visualized tones itself can improve the tones accuracy.

DISCUSSION

Reflection of the first results

With the results of our initial validation test, we can predict the answer to our research question. The preliminary test clearly illustrated that giving insight with visual feedback helped the user with the learning process with Chinese tones. To further assess the result of Tonal Star, a test plan has been written. A valid answer to the main question, therefore, can only be given once this test plan has been executed.

Regarding our hypothesis, we expected that giving visual feedback will simplify the learning of tonal languages. Filling the gap of not knowing the details of your performance; which is, with the currently available tools, based on the user's own interpretation.
Reflecting on the stated hypothesis, we as researchers believe the first results therefore verify the hypothesis. This however still needs to be proven completely using the further validation test.

Limitations of the current design and further improvements

Looking at the initial design of Tonal Star we think that there are still some improvements left to implement. Starting with the current method of visualization. Using the current spectrograms users are able to see the fluctuation of certain tones. These fluctuations however may be a bit too unclear for some users. One of the next steps, therefore, is improving the layout and design of the spectrogram. This can be done by separating or highlighting the different tones, making it more clear for the user on which tones he/she needs to improve. We think that with the use of machine learning, this can be automated.

Secondly, we think that the current design lacks personal experience. By the implementation of a large set of data from one user, a profile of the speaking habits of that user can be built. By analyzing the spectrograms using, for example, Convolutional Neural Networks, there can be determined how well the user pronounces the different tones and with what tones the user is still struggling. Using this data Tonal Star is able to give personal feedback to the users.

Lastly, we think a good addition to Tonal Star is an extended form into the physical world. Hereby embodying the tones and adding a sense of touch. The current design is solely based on an application on someone's tablet. That is why we, as researchers, think it is necessary to experiment with the implementation of Tonal Star in the form of a physical product. This could be in the form of a wearable or a gadget on the user's desk. Through the addition of a physical artifact, the user will be more engaged. Hereby adding a sense of though and giving a more embodied experience.

CONCLUSION

This research aimed to simplify the process of learning tonal languages. Based on the created functionalities around the visualization of the spoken tones, it can be concluded that the learning process improves by giving insight into the user's pronunciation. The research clearly illustrates that giving insight with visual feedback helps the learning process with Chinese tones, but it also raises the question of how applicable this is for other tonal languages. To understand the implications of these results better, future studies should focus on making it possible to directly give an indication about what tone and word combination is not pronounced well yet.

ACKNOWLEDGMENTS

We would like to thank Emilia Barakova, Jun Hu, Kostas Tsiakas and everyone involved in the Embodying intelligent behavior in social context course for providing us with weekly lectures and comprehensive feedback. We would also like to thank the other students in this course for their attention during our presentations and their feedback on our research.

REFERENCES

Aiken, M. (2019). An Updated Evaluation of Google Translate Accuracy. Studies in Linguistics and Literature, 3(3), p253. https://doi.org/10.22158/sll.v3n3p253

Dahl, G. E., Dong Yu, Li Deng, & Acero, A. (2012). Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. IEEE Transactions on Audio, Speech, and Language Processing, 20(1), 30–42. https://doi.org/10.1109/tasl.2011.2134090

Hao, Y.-C. (2012). Second language acquisition of Mandarin Chinese tones by tonal and non-tonal language speakers. Journal of Phonetics, 40(2), 269–279. https://doi.org/10.1016/j.wocn.2011.11.001

Hincks, R., & Edlund, J. (2009). PROMOTING INCREASED PITCH VARIATION IN ORAL PRESENTATIONS WITH TRANSIENT VISUAL FEEDBACK. Language Learning & Technology, 2009, 32–50. https://scholarspace.manoa.hawaii.edu/bitstream/10125/44190/13_03_hincksedlund.pdf

Kan, M. S., & Ito, A. (2020). Language Cognition and Pronunciation Training Using Applications. Future Internet, 12(3), 42. https://doi.org/10.3390/fi12030042

Li, X., Wenle, Z., Ning, Z., Chaoyang, L., Yongxin, L., Xiuwu, C., & Xiaoyan, Z. (2006). Mandarin Chinese Tone Recognition with an Artificial Neural Network. Journal of Otology, 1(1), 30–34. https://doi.org/10.1016/s1672-2930(06)50005-4

Matthew, S. (z.d.). WALS Online - Feature 13A: Tone. Wals. https://wals.info/feature/13A?s=20&z3=3000&z2=2999&z1=2998&tg_format=map&v1=cfff&v2=cf6f&v3=cd00#3/5.45/141.79

Rasul, S., Bukhsh, Q., & Batool, S. (2011). A study to analyze the effectiveness of audio visual aids in teaching learning process at university level. Procedia - Social and Behavioral Sciences, 28, 78–81. https://doi.org/10.1016/j.sbspro.2011.11.016

Settles, B., T. LaFlair, G., & Hagiwara, M. (2020). Machine Learning–Driven Language Assessment. Transactions of the Association for Computational Linguistics, 8, 247–263. https://doi.org/10.1162/tacl_a_00310

Tones in Mandarin Chinese. (2020). Misspandachinese. https://www.misspandachinese.com/resources/tones-in-chinese/

Wu, Y. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. Arxiv. https://arxiv.org/pdf/1609.08144.pdf

APPENDIX

Appendix I - Main program

```
import speechRecognition as vR
import speechRecord
import datetime
import plotSpectrogram as sP
from xpinyin import Pinyin

sentence = "我来自荷兰"
def compare_characters(original, inputs):
    length = len(original)
    if length <= len(inputs):
        formatted_input =
'{:{}}'.format(inputs, length)
    else:
        formatted_input =
'{:*<{}}'.format(inputs, length)
    count = 0
    accuracy = 0
    while count < length:
        if formatted_input[count] ==
original[count]:
            accuracy += 1
        count += 1
    return int(accuracy / length * 100)
#convert text into pinyin with tones as
numbers
def tone(words):
    p = Pinyin()
    tones =
p.get_pinyin(words,tone_marks='numbers')
    return tones
def extract_tones(pinyin):
    lists = pinyin.split('-')
    tones = []
    for i in lists:
        tones += i[-1]
    return tones
if __name__ == '__main__':
    r = speechRecord.Recorder()
    r.record()
    fileName =
datetime.date.today().strftime("%Y_%m_%d_%H:
%M:%S") + ".wav"
    r.save_wav(fileName)

    userInput = vR.recognize_voice(fileName)
    print(userInput)
    output = tone(userInput)
    print(output)
    print(extract_tones(output))
    scores = compare_characters(sentence,
userInput)
    if scores > 95:
        print("Congrats! You said it right!")
    else:
        print("Hmmm,you only said it {}%
right.\nTry again.".format(scores))
    sP.plot_speech(fileName)
    sP.plot.savefig('spec.png')
```

Appendix II - Speech recognition

```
import speech_recognition as sr
def recognize_voice(wav_path):
    r = sr.Recognizer()
    with sr.WavFile(wav_path) as source:
        audio = r.listen(source)
    try:
        return r.recognize_google(audio,
language='zh-cn')
    except sr.UnknownValueError:
        print("Google Speech Recognition could
not understand audio")
        return "error"
    except sr.RequestError as e:
        print("Could not request results from
Google Speech Recognition service;
{0}".format(e))
        return "error"
if __name__ == '__main__':
    results =
recognize_voice(wav_path="test.wav")
    print(results)
```

Appendix III - Speech record

```
from pyaudio import PyAudio, paInt16
import numpy as np
import wave
class Recorder:
    NUM_SAMPLES = 2000
    SAMPLING_RATE = 8000
    LEVEL = 500
    COUNT_NUM = 20
    SAVE_LENGTH = 8
    TIME_COUNT = 15
    Voice_String = []
    def save_wav(self, filename):
        wf = wave.open(filename, 'wb')
        wf.setnchannels(1)
        wf.setsampwidth(2) wf.setframer
ate(self.SAMPLING_RATE) wf.writeframes
(np.array(self.Voice_String).tobytes())
        wf.close()
    def record(self):
        pa = PyAudio()
        stream = pa.open(format=paInt16,
channels=1, rate=self.SAMPLING_RATE,
                         input=True,
frames_per_buffer=self.NUM_SAMPLES)
        save_count = 0
        save_buffer = []
        time_count = self.TIME_COUNT

        while True:
            time_count -= 1
            string_audio_data =
stream.read(self.NUM_SAMPLES)
            audio_data =
np.frombuffer(string_audio_data,
dtype=np.short)
            large_sample_count =
np.sum(audio_data > self.LEVEL)
            print("Recording...")
            if large_sample_count >
self.COUNT_NUM:
                save_count = self.SAVE_LENGTH
            else:
```

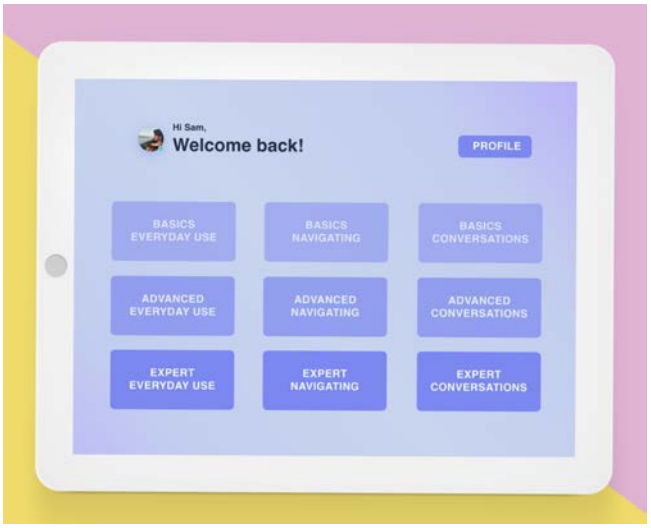
```
                save_count -= 1
            if save_count < 0:
                save_count = 0
            if save_count > 0:
                save_buffer.append(string_audi
o_data)
            else:
                if len(save_buffer) > 0:
                    self.Voice_String       =
save_buffer
                    print("Record
successfully.")
                    return True
            if time_count == 0:
                if len(save_buffer) > 0:
                    self.Voice_String       =
save_buffer
                    print("Record
successfully.")
                    return True
                else:
                    return False
if __name__ == "__main__":
    r = Recorder()
    r.record()
    r.save_wav("test.wav")
```

Appendix IV - Plot spectrogram

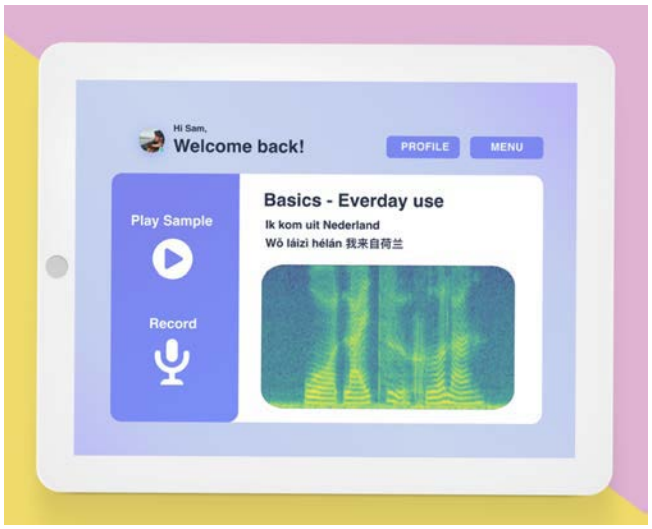
```
import matplotlib.pyplot as plot
from scipy.io import wavfile
from matplotlib.pyplot import figure
def plot_speech(file_name):
    figure(num=None, figsize=(10, 12), dpi=80,
facecolor='w', edgecolor='k')
    # Read the wav file (mono)
    sampling_frequency,     signal_data     =
wavfile.read(file_name)
    # Plot the signal read from wav file
    plot.subplot(211)
    plot.title('Spectrogram of a wav file')
    plot.specgram(signal_data,
Fs=sampling_frequency)
    plot.xlabel('Time')
    plot.ylabel('Frequency')
    plot.show()
if __name__ == '__main__':
    plot_speech("test.wav")
    plot.savefig('spec.png')
```

Appendix V - Prototype

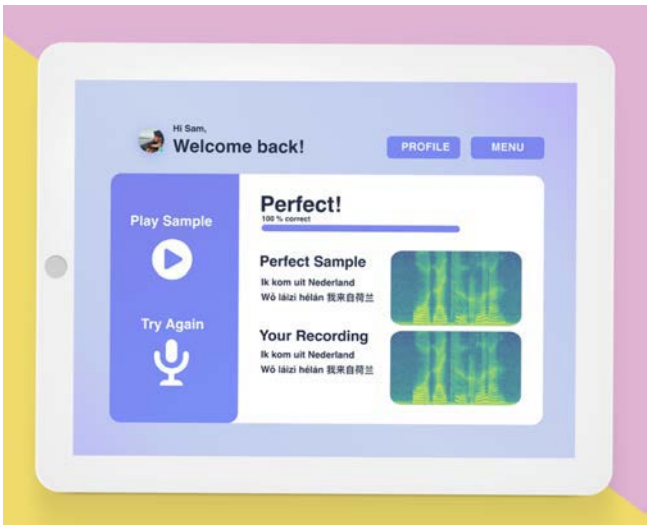
Logo of Tonal Star



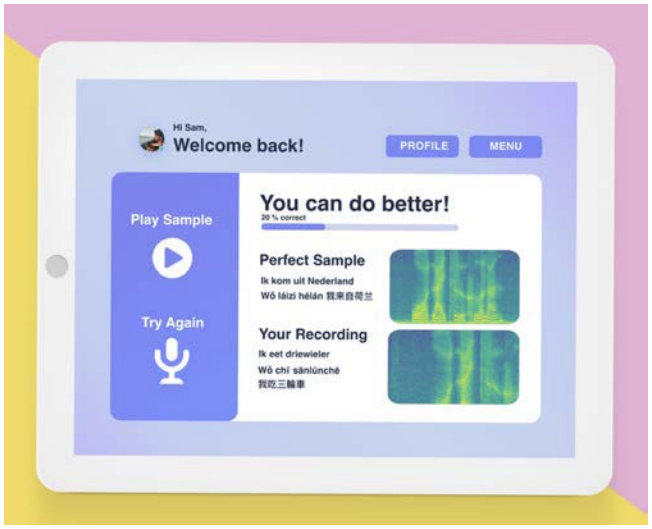
When the user log in on their personal tablet the app will show different exercises, the user is able to pick the one suiting their current skill level.



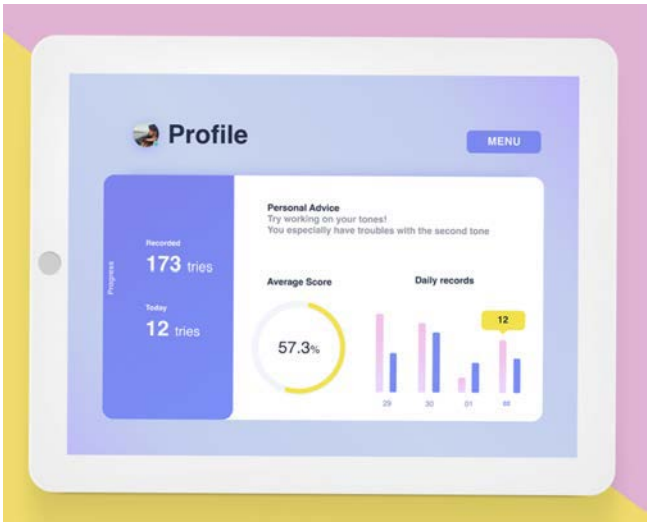
A new window will pop up giving the user a certain exercise, in this example they are asked to pronounce the translation of I am from the netherlands. The additional spectrogram gives the user visual feedback on the different tones in the sentence, The user is also able to hear a sample of right pronunciation. Now the user can press record to try and replicate the right pronunciation.



The user is encouraged to try again until a high score is achieved.



The Tonal Star app shows two spectrograms to visually compare the perfect pronunciation with the pronunciation of the user. Next to the spectrograms, the percentage in which the user's pronunciation corresponds to the correct pronunciation and the translation of the user's pronunciation are shown.



Appendix VI - Group cohesion workshop

Brainstorm session with coffee and cake, OFFLINE.