International Journal of Innovative Research in Engineering & Management (IJIREM)
ISSN: 2350-0557, Volume-9, Issue-5, October 2022
https://doi.org/10.55524/ijirem.2022.9.5.33
Article ID IJIR-1218, Pages 227-231
www.ijirem.org

An FPT.AI-Based Text-to-Speech Application's End-to-End Conversion Speed Analysis

M. Chaitanya Bharathi [1], Dr. A. Seshagiri Rao [2], and P. Ramalingamma[3]

[1,3] Assistant Professor, Department of Information Technology, PACE Institute of Technology and Sciences, Ongole, India
[2] Professor, Department of Information Technology, PACE Institute of Technology and Sciences, Ongole, India

Correspondence should be addressed to M.Chaitanya Bharathi; bharathi_m@pace.ac.in

ABSTRACT— In this paper, an FPT.AI-based text-to-speech (TTS) application is developed that converts Vietnamese text into spoken words. the applying is developed supported Django for Python ANd within the style of AN interactive website that is connected to an FPT.AI server through its application programming interface (API). the applying supports conversion of text to seven totally different Vietnamese speeches. Four out of seven voices are often wont to convert up to five hundred characters in an exceedingly single group action whereas the others support that of four hundred characters. supported the results obtained, the primary conversion time takes up to ten s to convert 400-character text into speech whereas the following times, given same text, it takes beneath one.8 s for the conversion. this is often applicable to any or all voices.

KEYWORDS- FPT.AI, TTS, performance, analysis, Vietnamese, voice

I. INTRODUCTION

In recent years, the event of machine-driven responding systems (i.e., chatbot, voicebot [1]–[4]) has enabled the in depth use of speech-to-text (STT) and text-to-speech (TTS) systems. The latter permits a system to talk out a given input text with somebody's voice [5], [6]. This provides a close-to-nature response to user UN agency interacts with the system. additionally, user experiences with the system may be considerably improved since the system will offer promptly responses to any requests. However, the standard of the responses depends on multiple factors like signaling quality, understanding of request and context, and conversion engine [4]–[8]. the foremost necessary one would be the engine.

There area unit many parameters which will be used for indicating the performance of a TTS system. the foremost common parameter is mean opinion score (MOS) that is loosely wont to live the naturalness of the generated speech [8]–[10]. However, this can be not enough to point the performance of a system from client perspective. the rationale is that end-users solely expertise end-to-end TTS conversion, therefore, although the core engine is incredibly quick, the intermediate communication media could have an effect on how briskly the system will perform the conversion. Thus, during this analysis, end-to-end conversion speed of AN FPT.AI-based TTS application is analyzed with the most concentrate on the connection between the length of the input text and its end-to-end conversion time.

The main contributions of this analysis are: (i) AN FPT.AI- primarily based TTS application exploitation Django for Python, (ii) performance analysis of the appliance for seven completely different supported voices and a number of other lengths of input text. The paper is organized as follows: Section II presents the analysis methodology; Section III discusses experimental results and analysis; and at last Section IV concludes this analysis.

II. METHODOLOGY

A. About FPT.AI API

In this work, FPT.AI API [11] is employed for interfacing between native host and remote FPT TTS server. This represents AN actual operating condition once AN external user has to use the API for changing text to speech. The TTS API takes four arguments for forming AN machine-readable text Transfer Protocol (HTTP) request before posting it to the server: (i) POST information that contain text to be reborn to speech; (ii-iv) voice, speed and prosody to create HTTP header. The latter 3 arguments aren't needed by the TTS server throughout an invitation. the rationale is that by default, traditional speed (speed =
0) and feminine voice (see Table I) (Thu Dung - Northern Female) area unit used. The prosody is just offered for question operating with male voice. Hence, during this work, it's not thought-about for performance analysis.

Table 1 presents the offered voices supported by FPT.AI engine. These voices area unit provided for trial users to convert up to ten,000 characters to speech monthly [11].

Table 1: Fpt.Ai Available Voices

No.	Voice	API Parameter
1	Ban Mai (Northern Female)	banmai
2	Le Minh (Northern Male)	leminh
3	Lan Nhi (Southern Female)	lannhi
4	Thu Dung (Northern Female)	female
5	Cao Chung (Northern Male)	male
6	Ha Tieu Mai (Southern Female)	hatieumai
7	Ngoc Lam (Hue's - Central Female)	ngoclam

For each request, a response will be returned to host application by the server. It has JavaScript Object Notation (JSON) format and contains a static HTTP link to download the converted audio file in *.mp3 format. In addition, the response has an error-or-success indicator plus a message detailing the system's feedback. Since, it may take some time for the server to generate the audio file, this work aims to assess the end-to-end conversion timing of the system.

B. Required Tools

In order to complete this work, the following tools are used

- Integrated Development Environment (IDE): Visual Studio Code version 1.39.2
- Programming Languages: hypertext markup language (HTML), cascading style sheets (CSS), Python.
- Framework: Bootstrap, Django for Python [12]

FPT.AI API [11].

Here, the IDE is employed for providing development atmosphere. 3 programming languages square measure utilized in conjunction to create the appliance. In particular: HTML is employed for developing web-based interface; CSS is employed for styling the interface; Python is core programing language that performs back-end operations. The Bootstrap framework helps to make the responsive interface whereas Django supports fast development of the online application. additionally, the FPT.AI engine performs conversion from text to speech and returns a reborn audio get into *.mp3 format.

B. Application Structure

In this analysis, the appliance structure is conferred in Fig. 1. Here, it's seen that there square measure 3 main input parameters that user will key into the system: text to convert to speech, the required speed of generated speech and voice kind. These inputs square measure fed into the online application developed supported Django for Python. This application is employed at native host.

There square measure four main files that square measure at the most vital to the appliance. The urls.py file situated in mysite folder is employed to initialize a localhost that address is http://127.0.0.1:8000/. once user runs the appliance, it'll either come AN address to admin folder containing administration data or synchronize with file urls.py situated in polls folder to attach server and consumer by their addresses.

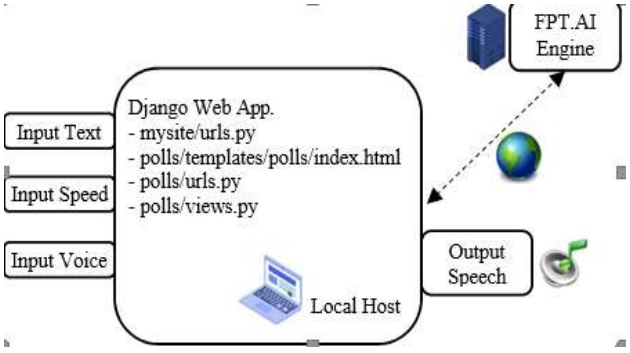

Figure 1: FPT.AI-based TTS application structure

The urls.py move into the polls folder imports path of native server and connects between views.py and index.html files by ways. Here, the views.py file contains the getVoice operate that performs all back-end tasks to complete the TTS conversion. On the opposite hand, the index.html file layouts the online application Associate in Nursingd provides an interface between native host and end-user.The workable content of the views.py file is bestowed within the Fig. 2. In Fig. 2, views.py is that the core controller. It defines a module containing all necessary functions for finishing a TTS task. during this file, the operate index(request) can perform rendering of the online interface given at index.html templet to get user's info like desired text, speech speed, and voice for conversion. These information ar keep into input_text, speech_voice, speech_speed variables severally. A link to FPT.AI server is made and keep into uniform resource locator. The human- machine interface address that the file can send the request to FPI.AI engine is https://api.fpt.ai/hmi/tts/v5. Next, Associate in Nursing machine-readable text transfer protocol header is created supported API key generated for the dedicated user, user's desired speech speed and voice.

```
from django.shortcuts import render
from django.http import HttpResponse
from django.http import HttpRequest
import requests

# Create your views here.
def index(request):
    return render(request, "polls/index.html")

def getVoice(request):
    input_text = request.POST["voice"]
    speech_voice = request.POST["name"]
    speech_speed = request.POST["speeds"]
    url = 'https://api.fpt.ai/hmi/tts/v5'
    header = {
        'api-key':
'BJtsE2z3CCpeRCixZL6EvVpjPmhCewHQ',
        'speed': speech_speed,
        'voice': speech_voice
    }
    response = requests.request('POST', url,
data=input_text.encode('utf-8'), headers=header)
    substr = response.text.split('"')[3]
#success
    context = {"link": substr}
return render(request, "polls/index.html",
context)
```

Figure 2: views.py content

Each registered user is allowed to use the TTS service for multiple requests amounting a complete of ten,000 characters monthly [11]. this is often comfortable for a replacement user to expertise this service from FPT. once an invitation is triggered, its response is keep in response variable. the appliance then checks if the response returns a hit state, it'll produce an occasion to update the appliance to playback the came speech.

B. measure Approach

With the look represented within the on top of section, user will expertise FPT.AI-based TTS services once registration to the system. However, so as to investigate the end-to-end conversion temporal order, the getVoice perform within the views.py file is emended as within the following algorithmic rule (see Fig. 3).

In Fig. 3, once the appliance starts, it captures startTime and receives from user text information, desired speech Speed, and Voice. the appliance then prepares header data to send TTS conversion request to the server. Before this action is performed, timeout variable is ready to zero.

When application receives response from FPT.AI server, it checks if character quota has exceeded, during this case, link context is ready to null as a result of no additional conversion is permissible. within the alternative case, if the response is succeeded, audio link string (urlstr) is extracted from the response. Next, the appliance checks if the link is obtainable on the server by causation a get request to that. If the link isn't prepared nonetheless, it continues to examine till ten s has passed, by then the response is taken into account as timeout. Here, it ought to be noted that the 10-s amount is often quite comfortable time for a user to attend for a response from a machine at that it doesn't cause negative experiences to the user. On the opposite hand, once the link is obtainable, it's passed to the link context. this is often accustomed update the data retrieved from server onto the online application (index.html). At this stage, the appliance obtains endTime. It writes log data for additional investigation if required. once timeout happens, rather than work interval (the distinction between endTime and startTime), a timeout text irecorded. Finally, the application triggers a rendering event to update the web interface and playback the retrieved speech.

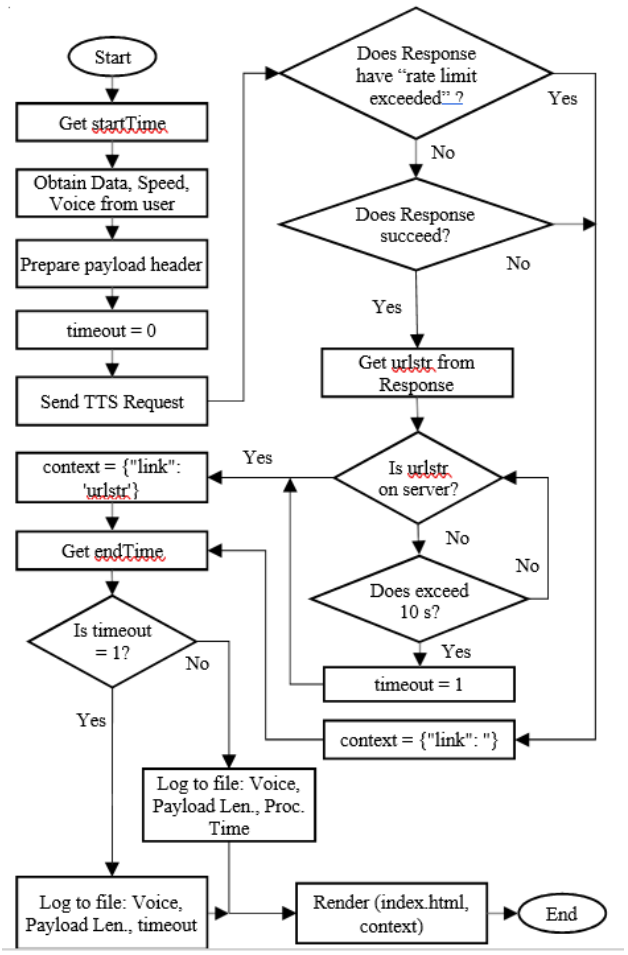

Figure 3: getVoice's modified algorithm

In order to assess the end-to-end conversion speed of the TTS engine, the experiment is carried out for all seven supported voices on an Intel Core i5-5250U system with 4 GB DDR3 RAM. The selected lengths of input text are: 10, 100, 200, 300, 400, and 500 characters. The least length typically represents a couple of spoken words while the most length is suitable for a shortspoken paragraphs. For representing randomness nature of TTS system, a random text for each length is obtained from a local news web page.

The measurement was performed for 10 times, with the same text for each length. The first conversion time represents randomness nature of TTS application while the second to the tenth times represent the nature of TTS application in which multiple users would like to convert the same text from a news page to speech for listening purpose. Furthermore, it is found that the TTS system will perform the conversion for a given text if it is new to the system and has not been recorded before. This is to save the conversion timing and enhance user experience.

III. RESULTS & ANALYSIS

As a result of the development, the application's web interface is presented in Fig. 4.

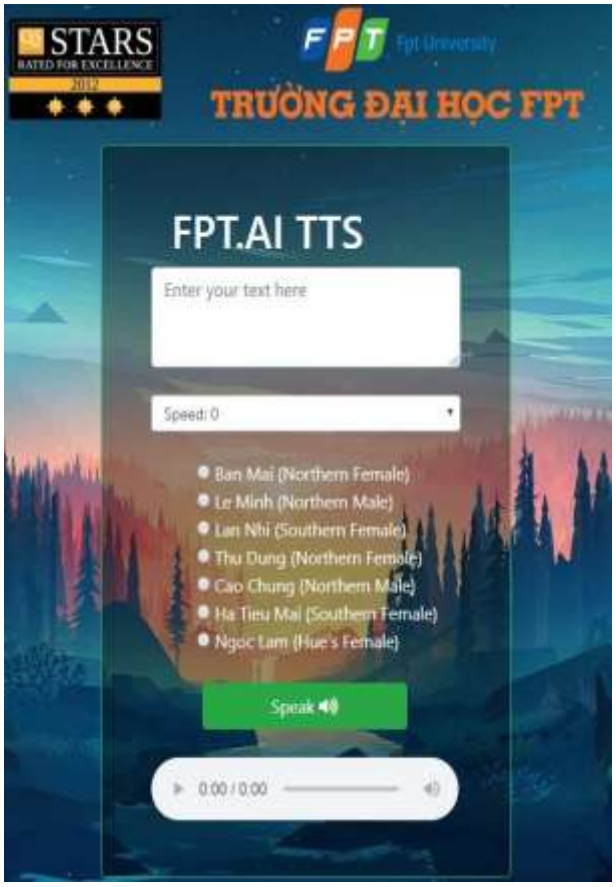

Figure 4: Application's web interface

In Fig. 4, it's seen that there square measure 3 main regions: (i) text space for writing a random text; (ii) configuration space for choosing speed of speech and a supported voice out of seven accessible voices; and (iii) user action space for triggering TTS conversion and playing-back the came speech. The back- finish of this application is processed by Python programing language and it's connected to FPT.AI core engine that permits up to ten,000-free-character-conversion for TTS application. There square measure seven accessible speeds of output speech starting from -3 to +3 with step size of one. once speed is zero, the speech is in traditional condition, not quick nor slow. once speed is -3 it's the slowest, and once speed is +3 it's the quickest. speed equaled to zero is additionally the default setting of the appliance.

Fig. five presents a graph of the end-to-end interval of the primary TTS conversion as a perform of input text length. In most cases, the primary interval plays a very important role in generating a recently keep speech of a given text. within the future requests of constant input text, the interval is often abundant smaller than that of the primary time. this could be determined by scrutiny figures five and half-dozen. this can be as a result of that given constant input text, the FPT.AI engine can generate constant output speech and store it into a static hypertext transfer protocol link. The generated file has *.mp3 format.

Therefore, the end-to-end conversion time within the future requests square measure usually the time to retrieve the generated audio file from the server that doesn't embody file generation time. In Fig. 5, it's seen that because the text length will increase, the time taken to perform TTS task will increase moreover. Out of seven voices, three voices particularly Ban Mai, Le Minh, Lan Nhi could not generate speech from 500-character input text. For the particular case of Lan Nhi, during the first request, at 400-character input, the system could not generate speech. The three voices also take more time to generate speech compared to the others. It takes about 10 s and 9 s correspondingly to generate a speech from 400 characters. For Lan Nhi, at most, it takes about 8.8 s to generate speech from 300 characters. Only Thu Dung voice has conversion speed proportional to the number of input characters.

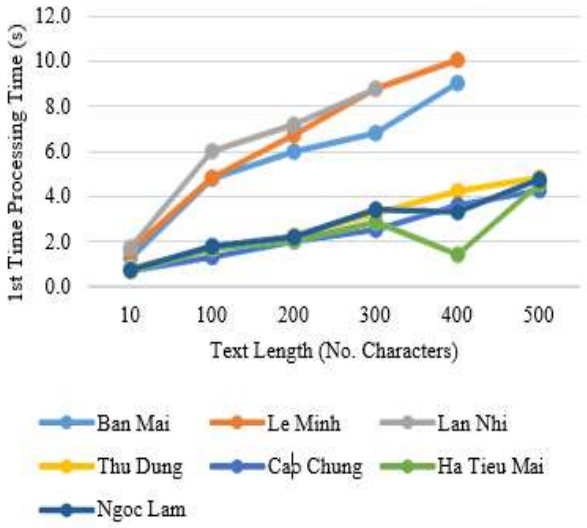

Figure 5: The first end-to-end conversion time vs text length

In order to further analyze the conversion speed of the developed engine, the average end-to-end conversion time vs text length is presented in Fig. 6.

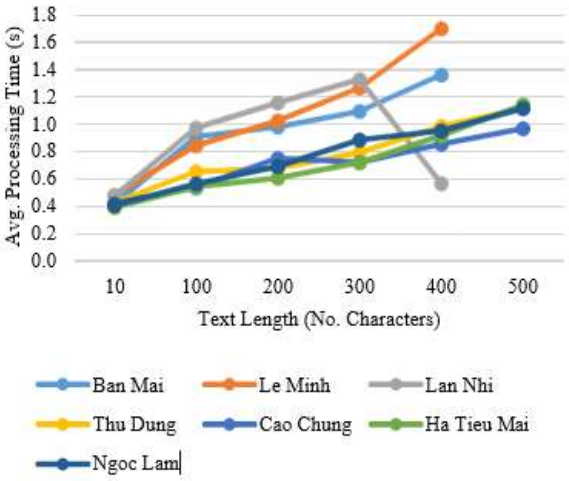

Figure 6: Average end-to-end conversion time vs text length.

Here the patterns for each voice are similar to those seen in Fig. 5. The difference is that on average, it takes less than 2 s to generate a speech from a given text with length up to 500 characters. For the case of 400-characters, Lan Nhi's average end-to-end conversion

time is 0.565 s. This indicates that although the first conversion could not be obtained (see Fig. 5, text length = 400 characters), the subsequent requests are still addressed properly. This information shall be useful for application developer to take into consideration in designing their TTS engine.

IV. CONCLUSIONS

This work has presented a development framework for processing text to generate different male and female voices for Vietnamese. Based on the results obtained, it is seen that as input text length increases, its end-to-end conversion time to obtained the converted speech also increases accordingly. It is important to note that TTS application can help people in various working and entertainment environment, especially with those having difficulties in communication. In future, we will further improve the developed application to provide more attractive graphical user interface, and improve its voice's quality.

REFERENCES

[1] B. Liu et al., "Content-Oriented User Modeling for Personalized Response Ranking in Chatbots," IEEE/ACM Trans. Audio Speech Lang. Process., 2018, doi: 10.1109/TASLP.2017.2763243.

[2] H. Cuayáhuitl et al., "Ensemble-based deep reinforcement learning for chatbots," Neurocomputing, 2019, doi: 10.1016/j.neucom.2019.08.007.

[3] S. Arsovski, H. Osipyan, M. I. Oladele, and A. D. Cheok, "Automatic knowledge extraction of any Chatbot from conversation," Expert Syst. Appl., 2019, doi: 10.1016/j.eswa.2019.07.014.

[4] D. C. Tran, H. S. Ha, and A. Khalyasmaa, "A Question Detection Algorithm for Text Analysis," in 2020 5th International Conference on Intelligent Information Technology (ICIIT), 2020, pp. 1–6.

[5] F. Eyben et al., "Unsupervised clustering of emotion and voice styles for expressive TTS," in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2012, doi: 10.1109/ICASSP.2012.6288797.

[6] W. Ping et al., "Deep Voice 3: 2000-Speaker Neural Text-to- Speech," in Proc. ICLR, 2018.

[7] D. C. Tran and A. K. M. K. A., "Effects of Soft-Masking Function on Spectrogram-based Instrument – Vocal Separation," in 2019 16th International Conference of the Pacific Association for Computational Linguistics (PACLING), 2019, pp. 1–5.

[8] J. Shen et al., "Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions," in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2018, doi: 10.1109/ICASSP.2018.8461368.

[9] D. S. Bormane, S. D. Shirbahadurkar, and U. D. Shiurka, "Performance of Marathi language TTS synthesis based on perceptual test and spectrogram analysis," in 2010 The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010, 2010, doi: 10.1109/ICCAE.2010.5451850.

[10] H. Tora and B. Uslu, "Naturalness analysis of the speech synthesized by a TTS card," 2016, doi: 10.1109/siu.2016.7496096.

[11] FPT, "FPT AI," 2019. [Online]. Available: https://fpt.ai.

[12] Django Software Foundation, "Django," 2019. [Online]. Available: https://www.djangoproject.com/.