Speech Synthesis System for Marathi Accent using FESTVOX

Sangramsing N.Kayte
Research Scholar
Department of Computer Science & IT,
Dr. Babasaheb Ambedkar Marathwada University,
Aurangabad, India

Monica Mundada
Research Scholar
Department of Computer Science & IT,
Dr. Babasaheb Ambedkar Marathwada University,
Aurangabad, India

Charansing Kayte, PhD
Assistant Professor
Department of Digital and Cyber Forensic
Aurangabad, Maharashtra, India

ABSTRACT
A Text To Speech synthesis (TTS) is the production of artificial speech by a machine for the given text as input. This field of study is known both as Speech Synthesis that is the "synthetic" (computer) generation of speech, and Text-To-Speech or TTS. It is the process of converting written text into speech. In the process of speech synthesis, mainly two processing components are used; they are NLP (natural language processing) and DSP (digital signal processing) modules. The speech synthesis has enormous applications such as reading for blind people, telecommunication services, language education, and aid to handicapped persons, talking books and toys, call center automation etc. The main aim of the project is to develop a TTS system producing a voice with Indian accent for the given input text. In this project, for the conversion of text to speech, we use Festival in Linux environment. Festival is a general pre-packaged tool for development of multi-language speech synthesis systems; and it will support most of the languages in the text to speech conversion. In this project, the speech generation process is done by using Festival frame work and speech tools. The voice model is generated by using festvox frame work, festival and speech tools. The required speech data for generating voice is recorded in noise less environment. The voice models can be generated by unit selection or clustergen modules present in festvox. It is observed from the generated voices that clustergen voices are better than unit selection voices.

Keywords
TTS, Festival, Festvox, speech syntheses.

1. INTRODUCTION
The festival is a speech synthesis system and it is developed in CSTR (Center for Speech Technology Research), university of Edinburgh. Festival is compatible to work with all types of voices and also in different platforms. It is having the open source license, so any one can use freely.

Festival Core system: The core system consists of the following features

Scheme-based scripting language: In order to easy representation of parameters and flow of control within the system, a scheme interpreter is provided as a command interpreter. This means much of festival‟s features are fully controllable at run time without having to re-compile the system offers both the advantages of a fast efficient language and the flexibility of an interpreted system.

General utterance representation: In C++ classes are used to offer a flexible and powerful representation for utterances. This is easy and efficient for writing functions using utterances. This is provided by the Edinburgh speech tools library.

Waveform I/O, formants, re-sampling: Many common waveform formats are cleanly supported so that waveforms, label files, coefficient files can easily be read and written. Re-sampling and changing formats is also supported making probability much easier.

Utterance, relations, and features, I/O: The utterance structure gives a common regular form to all utterances. Full support for access through scheme, and also in C++, it is made simple to use functions, utterances, or parts of utterances may be dumped to files in a human readable form for external manipulation and reloaded [20].

Standard data tools: A number of basic tools are available so you can easily use standard methods without having to build new tools. These include a viterbi decoder, N-gram support, regular expression, matching, linear regression support, CART support, weighted finite state transducers, and stochastic context free grammars.

Audio device access and spooling: The Edinburgh speech tools library offers direct and indirect support for many types of output audio device. Also spooling is supported, allowing synthesis to continue while playing a file.

Server/client model: client mode is provided so that a larger more powerful machine might be used as server remotely by smaller programs saving on both start up time, and resources required on the client end.

2. FESTIVAL ARCHITECTURE
A festival speech synthesis system consists of different modules and they all together produce the synthetic speech. The modules present in festival are:

Text Analysis and processing

Tokenization

Token identification

Token to word

Linguistic/prosody processing

Wave form generation

Render waveform

In festival framework utterance plays an important role in generation of synthetic speech. This framework takes an utterance and each of the modules present in it, manipulate in some way and pass on to the next module in it. Utterance consists of a set of items which are related through a set of

relations. Each relation consists of a list or tree of items. Items are used to represent objects like words or segments. Relations are used to link items together in a useful way. An item may have one or more relations [10].

2.1 Text analysis and processing

The first of the three major tasks in speech synthesis is the analysis of raw text into acceptable format that can be processed in a more reasonable manner. This section explains how to take arbitrary text and convert it into identifiable words chunked into reasonable sized utterances. The text analysis block takes the raw input text and produces the pronouncing format. Here all the abbreviations and numbers are expanded with respect to the context in the given text. It consists of three steps: Identifying tokens, Normalization of non-standard words (expansion of Tokens), Homograph disambiguation and chunk of utterance with sequence of pronouncing words[15][16][17].

2.2 Identifying tokens

The text is converted to tokens depending on the white spaces and punctuation marks. Whitespaces can be viewed as separators; Punctuation can also be separated from the raw tokens. Festival converts text into an ordered list of tokens, each with its own preceding whitespace and succeeding punctuation as features of the token.

2.3 Normalization of non-standard words:

In the given input text, all the words, which are available in the dictionary, are called standard words. Numbers, symbols, abbreviations and etc, which are not available in dictionary for their pronunciation, are called asnon-standard words. These words are converted into full pronunciation with the following stages.

Splitter: It will split the token not only white space but also with the punctuation.

Type identifier: It will identify the token type for expansion.

Token expander: Here the identified token is expanded depending on the context.

Language modeling: Language model is then used to select between possible alternative punctuations of the output.

2.4 Homograph disambiguation

In some languages like English some words are same but having different pronunciations, so here is the need of categorizing those words depending on the context in that sentence. This should be solved in text processing module.

2.5 Chunking into utterance

In festival, chunking tokens into utterances which are most reasonably recognized as sentences. Ideally chunks should be prosodic phrases In some languages festival may face some problems like tokenization without white space and number pronunciation to resolve these issues; festival will follow the letter to sound rules instead of lexicons to pronounce the words.

The following examples show how the raw text is converted into sequence of pronouncing words.

This is a laptop ->this is a laptop

He stole $100 from the bank -> He stole hundred dollars from the bank

He stole 1996 cattle on 25 Nov 1996 -> he stole One thousand Nine Hundred Ninety Six cattle on twenty fifth November Nineteen Ninety Six.

2.6 Linguistic/Prosodic processing

The second stage of festival speech synthesis system is linguistic and prosodic processing. The input for this stage is pronounceable words. To convert these pronounceable words in to segments with prosody, the system requires phones, durations and tune (F0 contour)[19]. The appropriate phone symbols for input words are extracted from the lexicons, which are available in dictionary. Lexicon consists of list of words with their corresponding phone symbols. The words which are not available in dictionary or lexicon list are follows the letter to sound rules to extract the phone symbols. Letter to sound rules are very difficult to write, but they are more powerful in the generation of phones from the words. These rules are generated with two ways. One is hand writing and another is building models using Classification and Regression Tree (CART) algorithm. Part of speech is tagged to the pronunciation words to remove the homograph disambiguation. It is necessary to split the utterances into prosodic phrases, to produce the natural sounding. This should be needed, because the people when speaks, they phrase their speech content depending upon, how they can express their sentence effectively and also how long their lungs are capable to contribute air in speech production. In festival, prosodic phrasing can be done by two methods, one is phrasing by decision tree and another is phrasing by statistical models. In phrasing by decision tree, a user defined scheme programmed file is used, in that, the rules to phrase the sentence are written. In phrasing by statistical models, system defined models are used, it can be done with viterbi decoding depending upon probability of breaks after the each word, its previous words and N-gram model [10]. Intonation, Durations and post-lexical rules all together called as prosody. Prosody plays an important role in the generation of natural speech as an output. Intonation is nothing but accent and F0 contour; these two parameters are extracted from the existing voice models. These intonational parameters decide the speaker and energy of the output speech[18].

Duration represents the length of the phone or unit present in the sentence. This can be measured by two ways in festival. One is making constant duration for the each phone and another one is taking the average duration of each phone that exists in the training speech database. Post-lexical rules are used to design good co-articulatory effects between the words. 3.2.3 Waveform synthesis This is the final and most important part of festival speech synthesis system. This receives phone information, prosody for synthesis from previous block and existed voice models. By combining all these parameters, it will produce synthetic speech as output. Depending on the voice models, the waveform synthesizer is differed to access the relevant and required information from voice models and produce synthetic speech.

2.7 Waveform synthesis

This is the final and most important part of festival speech synthesis system. This receives phone information, prosody for synthesis from previous block and existed voice models. By combining all these parameters, it will produce synthetic speech as output.

Depending on the voice models, the waveform synthesizer is differed to access the relevant and required information from voice models and produce synthetic speech.

3. SET INSTALLATION OF TOOLS FOR SPEECH SYNTHESIS SETTING UP SYSTEM

Linux (Ubuntu):

Install the ubuntu 12.10 or 13.04 version. Ubuntu is an operating system, one of the latest Linux distributions. It is open source software. After the installation of the ubuntu get all available updates from the internet. These updates automatically consist of basic compilers like shell, Perl and etc.

While updating ubuntu some of the packages like Bison, synaptic manager may not installs automatically but they are required. In this case install them from the ubuntu software center.

e.g. To install Bison, open ubuntu software center, type bison in search box, it search and display the package, then press install option or run the below command in the terminal.

"apt-get install bison"

Shared library tool (libssl0.9.8) is required to link the C/C++ functions in different projects, and then it will be installed manually.

To install shared library:

Open ubuntu software center, search and install libssl0.9.8 or run the below command in terminal

"apt-get install libssl0.9.8"

3.1 Installation of tools

Edinburgh Speech Tools provides a set of executables, which offer access to speech tools functionality in the form of a standalone program. As far as possible, the programs follow a common format in terms of assumptions, defaults and processing paradigms. Some of the common features of these tools are Arguments to functions can be specified in any order on the command line. Most programs can take multiple input files, which by default have no preceding argument. Output by default is to standard out. The -o flag can be used to specify an output file. Often programs can read many formats of input file. Wherever possible, this is done automatically. If this can't be done, the -itype flag can be used to specify the input file format. The output file format is specified by –otype fringe client manual section; Installation of festival speech synthesis system tool requires the sources of festival frame work and speech tools. Festvox project is another frame work to generate the new voice models from the recorded speech data base, developed in CMU (Carnegie Mellon University). To check whether the installation of festival is working or not, use the existed lexicons and voice models. Below mentioned release version tools and frame works are used for arctic speech data bases and its corresponding voice models. To generate voice models and use those models to synthesize the input text for Indic speech data base, current versions of festival, speech tools and festvox frame work are used.

Download the tools from the below website

(http://www.cstr.ed.ac.uk/downloads/festival/2.1/)

- festival-2.1-release.tar.gz
- speech _tools 2.4.tar.gz
- festvox-2.1-release.tar.gz
- festlex_CMU.tar.gz
- festlex_OALD.tar.gz
- festlex_POSLEX.tar.gz
- festvox_kallpc.tar.gz

Download the current version tools from the below website (http://festvox.org/11752/packed/)

- festival-2.1.1-current.tar.gz
- speech_tools-2.1.1-current.tar.gz
- festvox-2.5.3-current.tar.gz

Download ASR tools from cmu sphinx website given below (http://sourceforge.net/projects/cmusphinx/files/)

- sphinxbase-0.7
- sphinxtrain-0.9.1-beta
- sphinx2-0.6

Create a folder on Desktop and copy all the above downloaded tools and enter into the folder with root permissions.

Install all the tools which are downloaded above

1) Speech_tools

    1. tar –xvf speech_tools-2.1-release.tar.gz
    2. cd speech_tools
    3. ./configure
    4. make

If the errors occurs in speech_tools programming section while running „**make"**command, follow the below corrections.

- speech_tools/include/EST_Titerrator.h at line number: 212:7, 292:7 write **this->** and save.
- speech_tools/include/EST_TNamedEnum.h at line no 133:64 write **this->** and save.
- speech_tools/base_class/EST_Tsimplematrix.cc at line no 132:4, 130:11, 101:4 use **this->** before set_values, just_resize and also add **#include <string.h>** in the header file section because this program consist of **memcpy** function.
- speech_tools/base_class/EST_Tsimplevector.cc add **#include<string.h>** in the header files section because this program consist of **memset** function and at line no 74:7 use **this->** before just resize.
- After all the modifications are completed run "**make**" command.

After completion of this installation run „**make"** command. This completes installation of speech tools.

2) Festival

    1. tar –xvf festival-2.1-release.tar.gz
    2. cd festival
    3. ./configure

4. make

3) festvox

1. tar –xvf festvox-2.1-release.tar.gz
2. cd festvox
3. ./configure
4. Make

The same procedure is followed for the current version of speech tools, festival and festvox frame work.

3.2 Checking festival installation

Now copy the existing voice in to festival library which is downloaded earlier by extracting using below commands.

tar –xvf festlex_CMU.tar.gz

tar –xvf festlex_OALD.tar.gz

tar –xvf festvox_kallpc16k.tar.gz

While doing this make sure that cmu dictionary, lexicons and voices are copied into the festival library now the festival is ready to generate speech for the given input text with existing voice.

To run festival follow the below commands

UNIX shell commands:

echo "input text" | festival --tts

Now system plays the given input text

festival --tts file name followed by .txt

This command will play the text present in the given input file. Here the festival is the tool path and while playing text file, it should be in current directory.

3.3 Simple executing commands

festival> (voice. List) "press enter"

Here it will show the existing voice like
(bamu_mar_indic_cg)

festival>(voice_ bamu_mar_indic_cg) "press enter"

Now the festival is ready to play text with "ou_us_ram_indic" voice give the input to festival with the below command f

festival>(SayText " पुणे शहरातील एक मध्यवर्ती ठिकाण ")

festival>(tts "file name.txt" nil)

While running festival, no sound is playing and gives an error like "Linux can"t open

dev/dsp" then run the below command in home folder through terminal.

"printf ";use ALSA\n

(Parameter.set 'Audio_Method 'Audio_Command)\n

(Parameter.set 'Audio_Command "aplay -q -c 1 -t raw -f s16 -r $SR $FILE")\n"

>.festivalrc"

Up to now, installation of all the tools and the festival works with the existing voices which are downloaded from website and now trying to generate voices with some speech data base by using all the tools which are installed above.

4. CONCLUSION

In this Research paper gives the details about festival speech synthesis system. It explains about the core features of festival framework and text to speech conversion procedure in festival. It provides the details about required tools and frame works for speech synthesis system and their installation in ubuntu environment. It also gives testing procedure of festival speech synthesis system.

5. REFERENCES

[1] Ramani Boothalingam, V Sherlin Solomi, Anushiya Rachel Gladston, Lilly Christina, P Vijayalakshmi, Nagarajan Thangavelu, Hema A Murthy, "Development and Evaluation of Unit Selection and HMM-Based Speech Synthesis Systems for Tamil" 978-1-4673-5952-8/13/$31.00 c 2013 IEEE

[2] Samuel Thomas, "Natural Sounding Text-To-Speech Synthesis Based on Syllable-Like Units", ms thesis, Indian Institute of Technology, Madras, May-2007

[3] Paul Taylor, a text book on "Text to Speech Synthesis", University of Cambridge, United Kingdom

[4] T.Dutoit, "High-quality text-to-speech synthesis: an overview." Faculte Polytechnique de Mons, TCTS Lab, 31, bvd Dolez, B-7000 MONS (Belgium).

[5] Sami Lemmetty "Review of Speech Synthesis Technology" M.Tech., Helsinki University of Technology, Finland, 1999

[6] Amdal, T. Svendsen: "Unit Selection Synthesis Database Development Using Utterance Verification", Proc. Interspeech 2005, Lisbon, Portugal, Sept. 2005

[7] A.J. Hunt and A. Black: "Unit selection in a Concatenative speech synthesis system using a large speech database", Proc. ICASSP 1996, (Atlanta, USA), pp.373-376, 1996.

[8] Möbius: Corpus-based speech synthesis: Methods and challenges, Arbeitspapiere des Institutes für Machinelle Sprachverarbeitung, Univ. Stuttgart, AIMS 6 (4), pp. 87-116, 2000.

[9] Simon King, "A beginners' guide to statistical parametric speech synthesis" The Centre for Speech Technology Research, University of Edinburgh, UK

[10] A. Black, P. Taylor, and R. Caley, "The Festival speech synthesis system," http://festvox.org/festival, 1999.

[11] K. Prahallad, N. K. Elluru, V. Keri, S. Rajendran, and A. W. Black, "The IIIT-H Indic speech databases", in Proceedings of INTERSPEECH, Portland, Oregon, USA, 2012.

[12] Sri Rama Murty K, B. Yegnanarayana, Anand Joseph Xavier M, "Characterization of Glottal Activity From Speech Signals", IEEE Signal Processing Letters, vol. 16, no. 8, pp. 469-472, June 2009.

[13] A. Black and K. Lenzo, "Building voices in the Festival speech synthesis system," http://festvox.org/bsv/, 2000.

[14] Roman Timofe, "Classification and Regression Trees (CART) Theory and Applications" A Master Thesis, CASE, Berlin, December 20, 2004.

[15] Sangramsing Kayte, Monica Mundada "Study of Marathi Phones for Synthesis of Marathi Speech from Text" International Journal of Emerging Research in

Management &Technology ISSN: 2278-9359 (Volume-4, Issue-10) October 2015

[16] Sangramsing Kayte, Dr. Bharti Gawali "Marathi Speech Synthesis: A review" International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Volume: 3 Issue: 6 3708 – 3711

[17] Sangramsing Kayte, Monica Mundada, Santosh Gaikwad, Bharti Gawali "PERFORMANCE EVALUATION OF SPEECH SYNTHESIS TECHNIQUES FOR ENGLISH LANGUAGE " International Congress on Information and Communication Technology 9-10 October, 2015

[18] Sangramsing N.kayte "Marathi Isolated-Word Automatic Speech Recognition System based on Vector Quantization (VQ) approach" 101th Indian Science Congress Jammu University 03th Feb to 07 Feb 2014.

[19] Monica Mundada, Sangramsing Kayte "Classification of speech and its related fluency disorders Using KNN" ISSN2231-0096 Volume-4 Number-3 Sept 2014

[20] Monica Mundada, Bharti Gawali, Sangramsing Kayte "Recognition and classification of speech and its related fluency disorders" International Journal of Computer Science and Information Technologies (IJCSIT)

[21] http://tcts.fpms.ac.be/synthesis/introtts_old.html

[22] http://www.festvox.org/

[23] http://www.cstr.ed.ac.uk/

[24] http://en.wikipedia.org/wiki/Speech_synthesis

[25] http://hts.sp.nitech.ac.jp/

[26] http://festvox.org/11752/packed/

[27] http://audacity.sourceforge.net/

[28] http://www.speech.kth.se/wavesurfer/man.html