AN OPTIMIZED APPROACH TO VOICE TRANSLATION ON MOBILE PHONES

Nilesh N. Kapse[1], Kunal A Gole[2], Nikhil Dhuri[3]

[1]Computer Engineering Department, Yadavarao Tasgoankar Institute of Engineering and Technology
[2]Computer Engineering Department, Yadavarao Tasgoankar Institute of Engineering and Technology
[3]Computer Engineering Department, Yadavarao Tasgoankar Institute of Engineering and Technology

Abstract

Current voice translation tools and services use natural language understanding and natural language processing to convert words. However, these parsing methods concentrate more on capturing keywords and translating them, completely neglecting the considerable amount of processing time involved. In this paper, we are suggesting techniques that can optimize the processing time thereby increasing the throughput of voice translation services. Techniques like template matching, indexing frequently used words using probability search and session-based cache can considerably enhance processing times. More so, these factors become all the more important when we need to achieve real-time translation on mobile phones.

Keywords:-*Optimized voice translation, mobile client, speech recognition, language interpretation, template matching, probability search, session- based cache.*

---***---

1. INTRODUCTION

Language has always been the basis of any form of written or speech communication. However, the presence of multiple language and dialects has been a hindrance to effective communication. Especially in a nation like India where the language and dialect changes with region, the requirement of a middle translation layer that can eliminate the linguistic barriers becomes essential. Speakers from different regional identities should be able to interact with one another without the need to understand individual languages. Translation is performed in two streams: text-based translation and voice-based translation.

Text-based translation services mainly focus around capturing words and converting them to target language. However, voice based translation services have remained few and slow. This is because most of these models concentrate mainly on language interpretation and language generation. They fail to take into consideration the large amount of back-end processing that takes place while translation. Most translation methods make use of customized dictionaries to find the translated words. However, searching for relevant words and synonyms from such large dictionaries is slow and time-consuming. More so it also depends on the content of the sentence being translated.
In this paper, we propose an optimized approach to voice translation on mobile phones. This paper is aimed at mobile phone users who can then communicate with other users, irrespective of the other user's ability to understand the speaker's language. This paper can find varied applications in businesses, teaching and voice response systems.

This paper has been divided into several sections. In section I, we explain the complete system model. In the sub- sections of section II, we explain each component of the system in detail. Section III explains the working of speech recognition component. Section IV explains language interpretation and analysis module. Section V explains sentence generation module and section VI explains the text-to-speech synthesis module. In section VII, we state the applications of the system in various domains and sectors. In section VIII, we present the conclusion of our paper.

2. SYSTEM MODEL

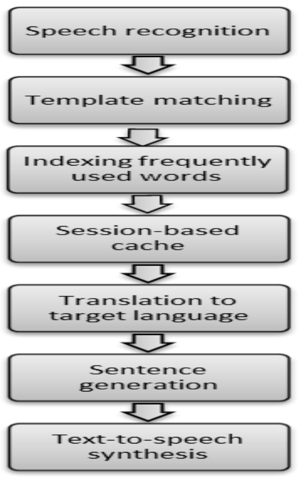

Fig1. System Model

The voice translation model consists of four main components namely: speech recognition, natural language interpretation and analysis, sentence generation and text-to-speech synthesis. The optimization is provided by the natural language interpretation and analysis module is which further divided into four parts namely: template matching, indexing frequently used words, session-based cache and translation to target language. Figure 1 depicts the overall system model for optimization of voice translation on mobile phones.

3. SPEECH RECOGNITION COMPONENT

The initiator dials a number on his mobile phone. This number is connected to a centralized server. The first phase is the speech recognition component. Speech recognition is the process of converting an acoustic signal captured by the mobile's microphone to a set of meaningful words. Speech recognition systems can be characterized by many parameters like speaking mode, speaking style, speaking accents and signal-to- noise ratio. An isolated word speech recognition system requires that the initiator on the mobile phone pause briefly between words, whereas a continuous speech recognition system does not. Speech recognition modules also take into consideration the speaking accents ofthe caller. Recognition is more difficult when vocabularies are large or have many similar-sounding words. When speech is produced in a sequence of words, language models or artificial grammars are used to restrict the combination of words.

3.1 Sphinx

To implement this speech recognition module, we use Sphinx 4[1], a speech recognition system developed at Carnegie Mellon University. Sphinx 4 is a refurbished version of the Sphinx engine which provides a more flexible framework for speech recognition, written entirely in the Java programming language. When a user calls from a mobile phone and speaks "I need to translate" on the microphone, the Sphinx module on the processing side captures the words from the audio form and converts it into text output.

Figure 2 shows the basic flow diagram of Sphinx 4 implementation [2].

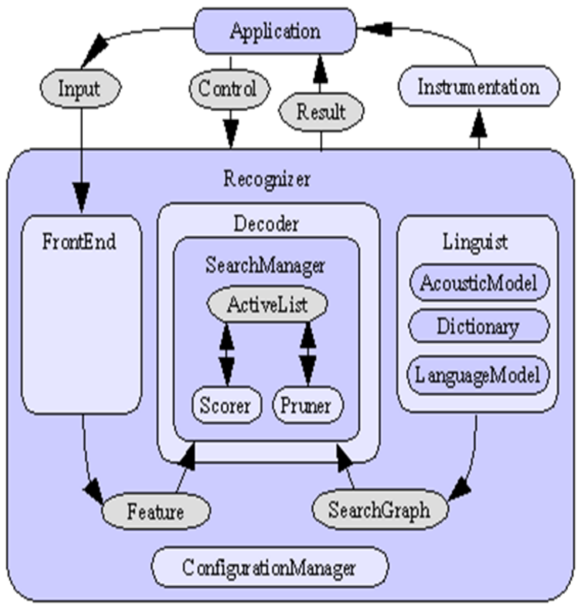

Fig 2: Basic Flow Diagram

The basic components of the Sphinx 4 model are as follows:
1. Input: The process starts with the voice input of the user, from the microphone of the mobile.
2. Configuration Manager: The configuration file is used to set all variables. These options are loaded by the configuration manager as the first step in any program.
3. Front End and feature: The front end is constructed, generating feature vectors from the input using the same process used during training.
4. Decoder: The decoder constructs the search manager which in turn initializes the scorer, pruner and active list. The SearchManager uses the Feature and the SearchGroup to find the best path fit.
5. Result: In the final step, the result is passed back to the application as a series of recognized words. Once the initial configuration is complete, the recognition process can repeat without re-initializing everything.

The performance parameters [2] for the speech recognition results are WER (Word error rate) and RT (Real time). WER signifies the percentage of errors committed during the recognition process. For command and control instructions, 5% WER (word error rate) is permitted. For a medium vocabulary 15% WER and for largevocabulary 30% WER is permissible. For a large vocabulary with short utterances, 50% WER is considered to be within an acceptable range. For a noisy environment with some accent, the thresholds are multiplied with a factor of 2.

Regression tests [3] performed on Sphinx-4 determine how it performs under a variety of tasks. These tasks and their results are as follows:

Isolated Digits (TI46): Runs Sphinx-4 with pre-recorded test data to gather performance metrics for recognizing just one word at a time. The vocabulary is merely the spoken digits from 0 through 9, with a single utterance containing just one digit.

Connected Digits (TIDIGITS): Extends the Isolated Digits test to recognize more than one word at a time (i.e., continuous speech). The vocabulary is merely the spoken digits from 0 through 9, with a single utterance containing a sequence of digits.

Small Vocabulary (AN4): Extends the vocabulary to approximately 100 words, with input data ranging from speaking words as well as spelling words out letter by letter. Medium Vocabulary (RM1): Extends the vocabulary to approximately 1,000 words. Medium Vocabulary (WSJ5K): Extends the vocabulary to approximately 5,000 words. Medium Vocabulary (WSJ20K): Extends the vocabulary to approximately 20,000 words.

Large Vocabulary (HUB4): Extends the vocabulary to approximately 64,000 words.

The following table states the performance of Sphinx-4:

Table 1: Performance of Sphinx4

Test	S4 WER	S4 RT (1)	S4 RT (2)	Vocabulary Size	Language Model
TI46	0.168	0.03	0.02	11	isolateddigits recognition
TIDIGITS	0.549	0.07	0.05	11	continuousdigits
AN4	1.192	0.25	0.20	79	trigram
RM1	2.88	0.50	0.41	1,000	trigram
WSJ5K	6.97	1.22	0.96	5,000	Trigram
HUB4	18.75	4.4	3.95	60,000	trigram

WER - Word error rate (%) (lower is better)
RT - Real Time - Ratio of processing time to audio time - (lower is better) S3.3 RT - Results for a single or dual CPU configuration
S4 RT(1) - Results on a single-CPU configuration
S4 RT(2) - Results for a dual-CPU configuration

This data was collected on a dual CPU UltraSPARC(R)-III running at 1015 MHz with 2G of memory.

4. LANGUAGE INTERPRETATION AND ANALYSIS

Natural language interpretation is sub-set of natural language processing. It involves comprehending the words in which context are they being referred to. This module breaks down the sentences into a machine-understandable form based on the vocabulary and the grammar of the language. We use WordNet® [3] which is a large English lexical database.The output obtained from the speech recognizer is given to the language interpretation module. This module is further divided into four sub-parts as follows:

4.1 Template Matching

Template matching checks the source input for commonly used phrases or sentences. Every language consists of a set of commonly spokenwords or sentences. Converting such sentences to the target language and replacing when required drastically reduces the processing time needed for translating the sentences.

Consider the statement "How are you?" as a commonly used sentence. The translated text output of this sentence is stored in a relational table. If a person speaks this sentence, then it will be directly translated instead of disassembling the words and analyzing them.

4.2 Indexing Frequently Used Words

The large lexical database of words will add to the time complexity of the process. The words are indexed based on the number of times a particular word has been used. The probability search algorithm is used to index the words in the database. In probability search the most probable element is brought at the beginning. When a key is found, it is swapped with the previous key. Thus if the key is accessed very often, it is brought at the beginning. Thus the most probable key is brought at the beginning. The efficiency of probability search increases as more and more words are being translated and indexed.

In order to measure the repetitiveness of words, we use hit ratio. If the word is found, then it produces a hit. If the word is not found, it is counted as miss. The ratio of number of hits is divided by the total references (hits and misses) is called hit ratio.

The orders in which the words get indexed depend on their hit ratios. Higher the hit ratio, higher is the probability of that word.

4.3 Pseudo Code for Probability Search

Input: Set of dictionary entries (n), Key to be searched (k)
Output: The searched key (k) Position of the key (i)

Algorithm:

```
Step i:  Search for the key till it is found
Step ii: If found then
Step iii: Store the value of key
Step iv: Swap key with its previous key
Step v:  Else
Step vi: Return null
```

The presence of a single while loop, makes the time complexity of this algorithm as O(n). The best case will be when the word to be searched is at the beginning while worst case will be when the word is at the end.

4.4 Session-Based Cache

The system maintains a session-based cache for each user requesting for the service. This works on the lines of a web cache which caches web pages. This is done to reduce to reduce bandwidth usage, server load, and perceived lag.

It is assumed that when a user engages in a conversation, there are bound to be multiple repetitions of certain words. Based on this assumption, we cache such words along with their translated text so that server processing time is saved.

4.5 Translation to Target Language

After the sentence passes through the first three phases of language interpretation and analysis, the final phase is translation to target language.

The words which are not translated by the template matching, frequently used words indexing and session-based cache are translated by this phase. As some of the words are already translated to the target language, the processing time required to translate the remaining words will be comparatively lesser.
We use Natural Language Toolkit [4], an open source software for performing sentence generation. Natural Language Toolkit is a suite of libraries and programs for carrying out symbolic and statistical natural language processing (NLP).

5. SENTENCE GENERATION

The collective set of translated words is converted to a meaningful sentence using sentence generation for the target language. Sentence generation is a natural language processing task of generating natural language from a logical form (set of translated words).

6. TEXT-TO-SPEECH SYNTHESIS

Text-to-speech synthesizer converts the sentence obtained from the sentence generation module into human speech form in the target language.

Figure 3 shows the overall architecture for FreeTTS.

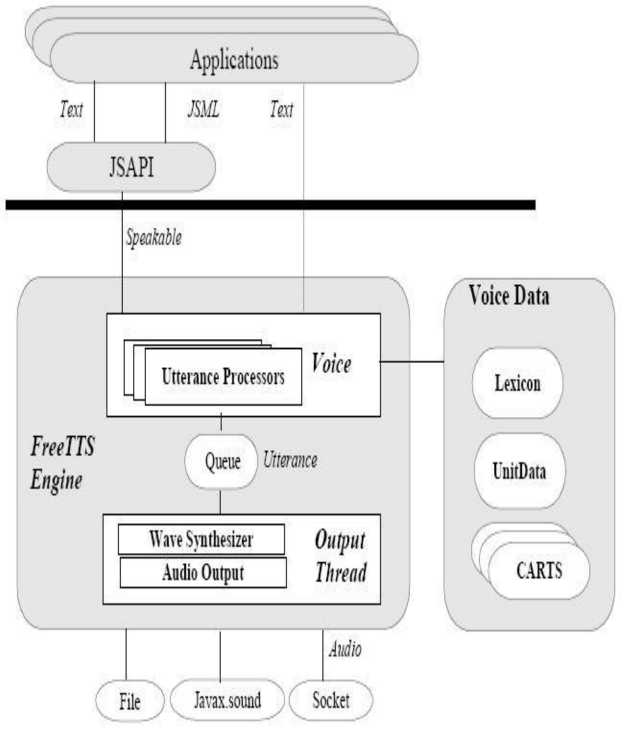

Fig 3: FreeTTS architecture

The core components of FreeTTS architecture [6] are as follows:
Voice Thread: Voice thread consists of a set of utterance processors. They perform the creation, processing, and annotation of an utterance structure.

Utterance processors: The utterance structure is a temporary object which is created by the voice for each audio wave generated by it. The voice initializes the utterance structure with the input text and then passes the utterance structure to a set of utterance processors in sequence. Each utterance processor adds additional items to the utterance structure in an hierarchical and relational manner. For example, one utterance processor creates a relation in the utterance structure consisting of items holding the words for the input text. Another utterance processor creates a relation that consists of items describing the syllables for the words, with each syllable item pointing back to the individual word items created by the other utterance processor.

Voice Data: Voice data sets are closely linked with the voice thread. These data sets are used by each of the utterance processors.

Output thread: The output thread is responsible for synthesizing an utterance into audio data and directing the converted au dio data to the appropriate audio playback mechanism.

7. APPLICATIONS

The most common application of voice translation is in the field of telephonic communication. By eliminating the dependence on languages, real-time communication can be performed without any common language backbone.

Other domains where voice translation can be used are education, entertainment, call center services and broadcast channels.

8. CONCLUSIONS

Language has always been a barrier to effective communication. As businesses expand and technology engulfs the entire globe, reliable and real-time translation becomes imperative. While considerable progress has been done in this direction, more efforts need to be taken in order to reduce the enormous processing time involved with it. With this paper, we propose a new system model to ensure effective real- time communication between two users who do not speak a common language while ensuring minimal computing time.

REFERENCES

[1]. CMU Sphinx 4, an open source speech recognition library. http://cmusphinx.sourceforgenet/sphinx4/
[2]. Performance parameters for Sphinx4 http://nsh.nexiwave.com/2009/08/how-to-improve-accuracy.html
[3]. Performance of Sphinx 4 http://cmusphinx.sourceforge.net/sphinx4/#speed_and_accuracy
[4]. CMU Sphinx 4 implementation http://cmusphinx.sourceforgenet/wiki/
[5]. WordNet an Electronic Lexical Database, MIT Press, ISBN: 026206197X
[6]. Natural language toolkit, a toolkit for natural language processing http://www.nltk.org/
[7]. FreeTTS 1.2 – Speech synthesizer system http://freettts.sourceforgenet/docs/imdex.php
[8]. FreeTTS - A Performance Case StudyBy: Willie Walker, Paul Lamere and Philip Kwok http://labs.oracle.com/techrep/2002/smli_tr-2002-114.pdf
[9]. Text-to-speech synthesis systems http://www.acapela-group.com/how-does-text-to-speech-work.html