Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 81 (2016) 188 – 193

Procedia Computer Science

ELSEVIER

5th Workshop on Spoken Language Technology for Under-resourced Languages, SLTU 2016, 9-12 May 2016, Yogyakarta, Indonesia

A Tool to Solve Sentence Segmentation Problem on Preparing Speech Database for Indonesian Text-to-speech System

Mohammad Teduh Uliniansyah[a,]*, Gunarso[a] Elvira Nurfadhilah[a], Lyla Ruslana Aini[a], Juliati Junde[a], Fara Ayuningtyas[a], Agung Santosa[a]

[a]*Center for Information and Communication Technology, BPPT, Puspiptek Serpong, Tangerang Selatan 15314, Indonesia*

Abstract

Creating a training data ready to be used for developing a text-to-speech (TTS) system can be a difficult task, since sometimes the recorded audio data is not the same with the prepared texts. To overcome differences between audio and text data, we developed a tool to segment audio data into sentences. As it is known, doing sentence segmentation of audio data manually needs efforts and resources. This paper presents a solution for alleviating problems encountered during segmentation process of audio data for developing an Indonesian TTS system. The tool was developed based on a fact that bahasa Indonesia is a syllable-timed language. We found that our tool reduces resources needed for segmenting Indonesian audio data.

Peer-review under responsibility of the Organizing Committee of SLTU 2016

Keywords: training data; TTS; segmenting audio data; Bahasa Indonesia; Syllable-timed

1. Introduction

As it is known, usually intelligibility and naturalness are two main points used to judge whether a TTS system is good or not. In order to maintain naturalness, a text corpus containing sentences from a novel is created, since usually a novel contains many conversational sentences. An Indonesian novel was chosen since there are many conversational sentences, apart from the fact that it was famous. Sentences are segmented by using an Indonesian

* Corresponding author. Tel.: +6-221-757-91260; fax: +6-221-757-91284.
E-mail address: teduh.uliniansyah@bppt.go.id

Peer-review under responsibility of the Organizing Committee of SLTU 2016
doi:10.1016/j.procs.2016.04.048

grammatical rule stating that a sentence boundary is marked by a period/exclamation mark/question mark (which can be followed by a double quote), followed by a word with capital letter as its 1st letter[1].

A speaker was then elected through an audition. Several speaker candidates were asked to say several sentences, and his/her voice clarity was judged using Perisalah, an Indonesian speech-to-text system that we developed earlier. However, during recording, the elected speaker sometimes combined two or more adjacent sentences, or skipped one or more words/sentences. These occurred especially for those of conversational sentences, resulting in differences between audio data and prepared texts. This creates problems on segmenting audio data, since it was not segmented manually during recording in order to preserve naturalness of speaker's voice.

Recently, many researches on audio data segmentation used hidden Markov model (HMM), conditional random field (CRF), etc. However, these advanced methods require labeled audio data for training that we do not have. Therefore, to address problems previously mentioned, we utilize a simple yet effective method.

Firstly, we tried to segment the audio data automatically into sentences by using available tools such as Audacity, Ffmpeg, and sox. Nevertheless, the results were not accurate, so that we decided to develop a tool to segment the audio data. The tool segments audio data was developed based on a fact that bahasa Indonesia is a syllable-timed language[2]. Experiments show that our tool can reduce resources needed to segment audio data.

Organization of this paper is as follows: chapter 1 briefly talks about background, problems, and method to overcome the problems, chapter 2 discusses related works, chapter 3 presents experimental data, chapter 4 talks about segmentation tool that we developed, chapter 5 presents experiment results, and chapter 6 gives conclusions of our research.

2. Related works

The process of identifying boundaries between sentences in spoken natural language is challenging, and numerous methods have been developed to do this task. Recent researches used various modeling techniques such as: prosody, syllable-unit, n-gram language, silent duration, and conditional random field (CRF).

A study of prosody-based automatic segmentation done by Shriberg et al.[3] described techniques using decision tree and hidden Markov modeling. They also combined prosodic cues with word-based approaches. Experiment results showed that the prosodic model alone performed on par with, or better than, word-based statistical language models and obtained a significant improvement over word-only models using a probabilistic combination of prosodic and lexical information across text and corpora.

Gotoh et al.[4] developed two statistical models: n-gram type language model and pause duration model to estimate sentence boundary from audio data. The pause duration model outperformed the language modeling approach, and the combination of the two gave best result.

In contrast to previous methods, conditional random fields (CRF) is a model that was originally developed in the segmentation of sentences in the text. CRF has been successfully used for text processing, but has not been developed for the segmentation of the sentence in the speech. Based on research by Liu et al.[5], CRF models yielded a lower error rate than the HMM and Maximum entropy models on sentence boundary detection in speech. The best results are achieved by three-way voting among the classifier.

All of the above methods require a labelled training data, a resource that we do not have. There are alternative methods based on syllable counting that do not require training data.

Many techniques have been developed to count syllables for any sentence that could be used for the purpose of speech rate determination. These techniques are usually done by detecting vowel positions in the syllable. Dekens et al.[6] has developed Low frequency Modulated Energy (LFME) algorithm as an improvement in speech rate algorithm. The LFME results have shown that it can reduce RMS errors and increase correlation coefficient.

Our work uses a yet simpler method by counting the syllable utterance time. Utilizing the fact that bahasa Indonesia is a syllable-timed language and the speech data processed is taken from a single speaker. So, we expected that the average syllable utterance time has a very small variance.

3. Experimental data

As mentioned previously, we conducted an audition to choose a speaker. Several speaker candidates were asked to say several sentences, and his/her voice clarity was judged using Perisalah, an Indonesian speech-to-text system that we developed earlier. A text corpus is created to be read by the speaker. The text corpus contained 5,000 sentences taken from a popular novel. Sentence alignment was done by following a grammatical rule saying that a sentence boundary is marked by a period/exclamation mark/question mark (which can be followed by a double quote), followed by a word with capital letter as its 1st letter. The following table gives detailed data of the text corpus:

Table 1. Text corpus.

Items	Count
Number of sentences	5,000
Number of unique sentences	4,957
Number of words	40,993
Number of unique words	5,805
Number of monophones	215,771
Number of distinct monophones	31
Number of biphones	210,773
Number of distinct biphones	669
Number of triphones	205,775
Number of distinct triphones	5,474

The recording process used following specification to create audio data: sampling rate 16 kHz and data size 16 bit. The resulting audio data has length approximately 7.25 hours.

4. Methods

The following is the summary of the method that we used to approximate sentence boundary. Using the fact that bahasa Indonesia is a syllable-timed language and each audio data to be segmented is recorded from one individual, then the average time to utter a syllable can be computed (t). Using t, the time required to utter each sentence can be approximated (T_i) by multiplying t with the syllable count of the sentence (N_i):

$$T_i = t \cdot N_i \tag{1}$$

The computed T_i and a detected paused near T_i is then used to get the possible sentence boundary.

Syllable count of each sentence (N_i) is calculated from the text data. First, the text data is normalized by expanding it into its equivalent phonetic notation. For syllable count purposes, the abbreviation expansion and diphthong detection are very important. After the normalization, N_i can be acquired by calculating the total number of vowels and diphthongs in each sentence.

To get t, first we segment the audio data by only taking the non-silent segments. All segments that only contain noise are removed. The total time of resulting cleaned segments is then calculated (T_a). The total count of the syllable (N_a) is computed from the text data used for recording. T_a is then divided by N_a to get the average syllable utterance time t.

Since the segments are separated by silence, the approximation of sentence boundary can be detected by sequentially adding segment into a segment accumulator, and when the total length of the accumulated segment is close to T_i then the last added segment is assumed to be the sentence boundary. Since this process is a sequential process, an error made in current step will propagate to the future steps. This fact makes the whole process hard to be done automatically. Because of this, a semi-automatic approach is selected. A GUI based application is designed to enable human intervention when an error is detected by the automatic segmentation procedures.

The GUI requires the following as inputs: non-silent segments of audio data including information about its length and the silence length to next segment, and syllable count of each sentence. Using the GUI, the automatic boundary search will be run, when an error is detected, the operator must make amend to it. The error is detected by calculating a cost value (C_i) and also considering the distance (D_i) between the detected boundary segment to the next non-silence segment. If C_i is greater than a threshold (*TC*) or *Di* is greater than a threshold (*TD*), then an error event is triggered. C_i is calculated by dividing absolute difference of T_i and the length of the accumulated segment (T_{si}) with T_{si}:

$$C_i := |T_i - T_{si}| / T_{si} \tag{2}$$

Upon receiving error notification, the operator must find the first sentence that has an alignment error. The alignment error can be fixed by removing last segment, adding next segment, forcing validation or, in some extreme cases, editing the sentence. The GUI provides a function to see the text of any sentence and a function to play any audio segments. After the alignment error is fixed, the automatic alignment proses can be rerun. This procedure is repeated until all alignments are completed without error.

5. Experiment results

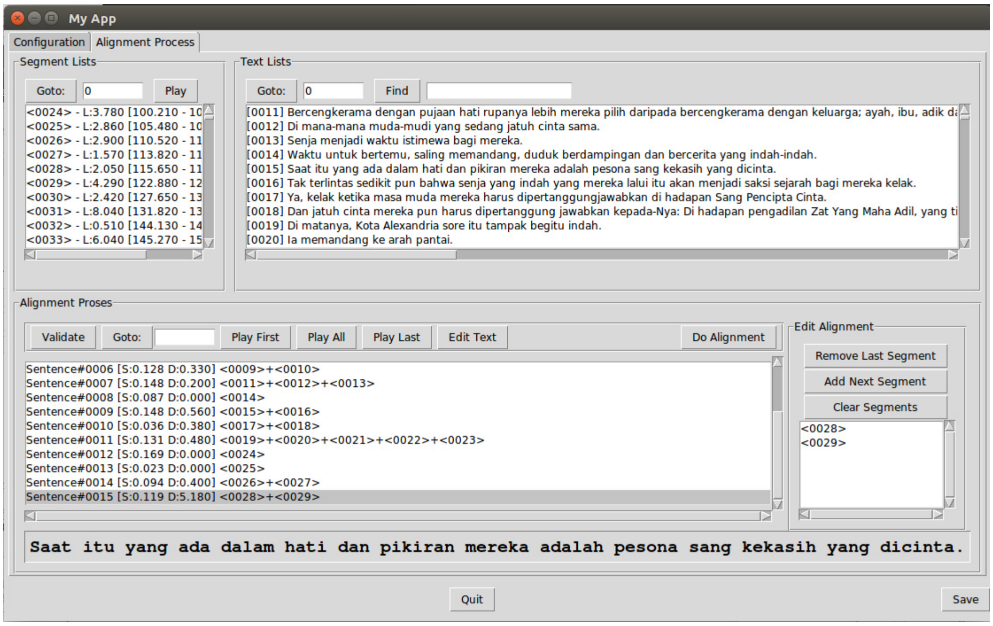
Prior to using the GUI, speech data and its text script must be preprocessed to generate some additional data. The speech data was segmented using Audacity to get a list of non-silence segments. A value used for minimum silent gap is set small enough to make sure that all targeted sentence are segmented. After taking the distribution of detected silent gaps from the speech data, 0.3 s was selected as the minimal silence gap. The list generated by Audacity consists of the start and stop of each non-silent segment.

From this non-silent segment list, we marked all segments that only contain noise. The total length of these non-silent segments minus all the noise segments were added together to get T_a.

From the text data, each sentence was normalized into its phonetic pronunciation using a dictionary lookup procedure. Vowels and diphthongs were then counted from the phonetic pronunciation representation of each sentence to approximate the total syllable in each sentence, N_i. After that, N_a was calculated by summing up N_i. From the acquired T_a and N_a, the average syllable time *t* is calculated as:

$$t := N_a / T_a \tag{3}$$

Speech data, its text script, non-silent segment list, *t* and list of N_i were then used as input for GUI. A screen shot of the GUI is shown in Fig. 1.



```
<0024> - L:3.780 [100.210 - 10
<0025> - L:2.860 [105.480 - 10
<0026> - L:2.900 [110.520 - 11
<0027> - L:1.570 [113.820 - 11
<0028> - L:2.050 [115.650 - 11
<0029> - L:4.290 [122.880 - 12
<0030> - L:2.420 [127.650 - 13
<0031> - L:8.040 [131.820 - 13
<0032> - L:0.510 [144.130 - 14
<0033> - L:6.040 [145.270 - 15
```

```
[0011] Bercengkerama dengan pujaan hati rupanya lebih mereka pilih daripada bercengkerama dengan keluarga; ayah, ibu, adik da
[0012] Di mana-mana muda-mudi yang sedang jatuh cinta sama.
[0013] Senja menjadi waktu istimewa bagi mereka.
[0014] Waktu untuk bertemu, saling memandang, duduk berdampingan dan bercerita yang indah-indah.
[0015] Saat itu yang ada dalam hati dan pikiran mereka adalah pesona sang kekasih yang dicinta.
[0016] Tak terlintas sedikit pun bahwa senja yang indah yang mereka lalui itu akan menjadi saksi sejarah bagi mereka kelak.
[0017] Ya, kelak ketika masa muda mereka harus dipertanggungjawabkan di hadapan Sang Pencipta Cinta.
[0018] Dan jatuh cinta mereka pun harus dipertanggung jawabkan kepada-Nya: Di hadapan pengadilan Zat Yang Maha Adil, yang ti
[0019] Di matanya, Kota Alexandria sore itu tampak begitu indah.
[0020] Ia memandang ke arah pantai.
```

```
Sentence#0006 [S:0.128 D:0.330] <0009>+<0010>
Sentence#0007 [S:0.148 D:0.200] <0011>+<0012>+<0013>
Sentence#0008 [S:0.087 D:0.000] <0014>
Sentence#0009 [S:0.148 D:0.560] <0015>+<0016>
Sentence#0010 [S:0.036 D:0.380] <0017>+<0018>
Sentence#0011 [S:0.131 D:0.480] <0019>+<0020>+<0021>+<0022>+<0023>
Sentence#0012 [S:0.169 D:0.000] <0024>
Sentence#0013 [S:0.023 D:0.000] <0025>
Sentence#0014 [S:0.094 D:0.400] <0026>+<0027>
Sentence#0015 [S:0.119 D:5.180] <0028>+<0029>
```

```
<0028>
<0029>
```

```
Saat itu yang ada dalam hati dan pikiran mereka adalah pesona sang kekasih yang dicinta.
```

Fig. 1. GUI of audio data segmentation tool.

Using the data mentioned in chapter 3, we have experimented by letting three participants conducting a segmentation process using our GUI. Table 2 shows the results of the experiments.

To measure effectiveness of our tool, we conducted an experiment with three participants only due to resource limitations. Table 2 shows that the three participants completed experiments within similar time period. The results show that using our tool can greatly reduce the time needed to do the segmentation compared to manual approach. Manual segmentation will at least requires the operator to listen to the whole audio data, which in our case, this can take more than 7.25 hours.

Table 2. Segmentation results done by three participants.

Item	Participant A	Participant B	Participant C
Time Needed (minutes)	140	180	190
Manual Correction	208	208	209
Manual Validation	63	63	63
Text Correction	41	41	41

‘Time needed’ is the time needed to complete segmentation process using our GUI for the whole 5,000 sentences. 'Manual correction' is number of how many times the operator needed to add next segment to a sentence or remove incorrectly added segment from a sentence 'Manual validation' is number of how many times the operator must manually validated an alignment that triggered an error event. 'Text correction' is number of how many times the operator must manually edit sentence. From the experiments, we found following errors in the recorded speech data:

- A sentence was read as two separate sentences.
- Two or more sentences were read as one sentence.

To test the validity and correctness of the result of the segmentation process, we conducted a statistical validity check using 99% confidence level and 3% confidence interval. The test was done by randomly sampling 1,350 samples from the segmentation result and checking manually the validity of the alignment. We found an error of 2.9% ± 3% from the sampling data.

Using the validated alignment result, we calculated actual distribution of average time needed to utter a syllable for each sentence. We found the following distribution function as shown in Fig. 2. The distribution resembled a Gaussian distribution with average of 186.7, variance of 36.19, and mode 180.

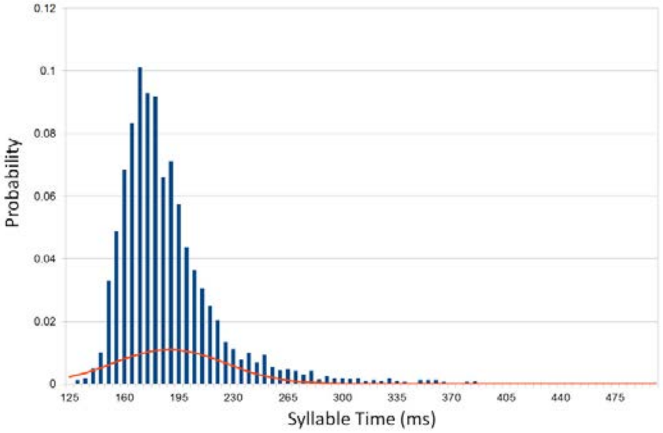


Fig. 2. Graph of syllable time distribution.

6. Conclusions

We have developed a GUI to segment audio data from a single speaker into sentences. The segmentation process is done semi-automatically using approximation of sentence utterance time, which is calculated from average syllable-time multiplied by the number of syllable in the sentence. The GUI reduced the resource requirement for the segmentation process by accelerating the sentence alignment process. The results of the segmentation show that there is still an error of 2.9% ± 3% with confidence level of 99%. In the future, we would like to use the syllable counting methods to enhance the result of the segmentation process.

References

1. H. Alwi, S. Dardjowidjojo, H. Lapoliwa, A. Moeliono. *Tata Bahasa Baku Bahasa Indonesia*. Third Edition. Jakarta: Balai Pustaka; 2003.
2. R. M. Dauer. Stress-timing and syllable-timing reanalyzed. *Journal of Phonetics, Vol 11(1)*; Jan 1983. p. 51-62.
3. Shriberg, Elizabeth, A. Stolcke, D. Hakkani-Tür, and G. Tür. Prosody-based automatic segmentation of speech into sentences and topics. *Speech communication 32, no. 1*; 2000. p.127-154.
4. Gotoh, Yoshihiko, and S. Renals. Sentence boundary detection in broadcast speech transcripts. 2000.
5. Y. Liu, S. Andreas, S. Elizabeth, and H. Mary. Using conditional random fields for sentence boundary detection in speech. *In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics; 2005. p. 451-458.
6. Dekens, Tomas, H. Martens, G. V. Nuffelen, M. D. Bodt, and W. Verhelst. Speech rate determination by vowel detection on the modulated energy envelope. *In Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European, IEEE*; 2014. p. 1252-1256.