Automatic Children's Reading Tutor on Hand-Held Devices

Xiaolong Li, Li Deng, Yun-Cheng Ju and Alex Acero

Microsoft Research, One Microsoft Way, Redmond, WA 98052
{xiaolli,deng,yuncj,alexac}@microsoft.com

Abstract

This paper presents an Automatic Reading Tutoring (ART) system using state-of-the-art speech recognition technologies aimed to improve children's oral reading ability. The features of this system include a compact and robust language model designed for detecting disfluencies in children's speech, low-footprint implementation, and built-in microphone array. Our system is targeting on hand-held devices to provide better accessibility, flexibility, and freedom for children's reading practice. The focus of this paper is on the current system's architecture, which has achieved real-time performance on two hand-held, small-form-factor devices (UMPC and Motion Tablet), with the same detection rate and false alarm rate as on desktop PCs. We also report the latest effort on a prototype system running on a PDA (Windows Mobile 6).

Index Terms: Automatic Reading Tutoring (ART) system, children' speech recognition, hand-held devices

1. Introduction

An Automatic Reading Tutoring (ART) system is an interactive tool using speech recognition technology to help children improve oral reading ability [1-4]. The motivation of developing ART systems is based on the facts that 1) the children's reading ability is extremely important for their success, and it is a crucial measurement for a nation's education and literacy level; 2) in both developed and developing countries teachers are insufficient in providing effective reading tutoring for young children [5,6]; 3) the significant progress of automatic speech recognition technology during the last two decades provides a possible solution of building a computer-based reading tutor to save both cost and time for children's literacy-building. Based on the state-of-the-art speech recognition technologies, an ART system is aiming to provide teacher-similar tutoring ability in children's phonemic awareness, vocabulary building, word comprehension, and fluent reading. The basic functions of an ART system include three aspects [7]: 1) Tracking --- it will track children's oral reading against the story text in real-time speed, give children feedbacks to show the current word location, and detect any reading miscues including stops, pauses, mispronunciations, and partial pronunciations; 2) Scaffolding --- it will provide on-line help information, in either a passive or an active way, to teach children how to pronounce a word, read a whole sentence, or explain the meaning of a specific word; and 3) Profiling --- it will measure and report the learning progress including reading fluency level, new vocabulary learned, and testing scores, etc.

There have been different ART systems reported including some commercial products, such as CMU's Project LISTEN [1], Univ. of Colorado-Boulder's Literacy Tutor [2], Soliloquy Learning's Reading Assistant [3], etc. Some field testing from those systems have shown that ART systems can improve children's reading ability more effectively than regular class-room studies [8], and with a faster speed [1].

Those previously reported systems are mainly targeting on desktop-based scenarios where children still need to sit in front of the desktop computers, wear a headset, and use the system under the watch of teachers or parents as in regular classroom studies. On one hand, this scenario has limited time for children's reading practice. Also especially for young children from five to seven years old, it is difficult for them to operate a desktop computer without guidance from teachers or parents. On the other hand, from our usability study, we find that the use of headset usually causes difficulties for children since they tend to play with it and it easily adds noise and distortions into the speech signals due to common misplacement and movement of the headset. To extend these limitations, we are building an ART system which targets on hand-held devices with a stylus and built-in microphone array. With such an un-tethered ART system, children will have better accessibility and flexibility in practice, and may have more freedom and a greater degree of personalization for themselves instead of being watched by their parents or teachers all the time. It is also easier for children to use a stylus than mouse/keyboard to operate a device, as shown by some popular game devices such as Leap Frog's Leapster [14] and Nintendo DS [15]. In other words, the user experience will be much improved by our hand-held ART system.

In this paper for the first time we present initial results for our ART system which has achieved real-time performance on two small form-factor hand-held devices running Windows XP, i.e., Motion LS 800 Tablet PC [16], and Samsung Q1 UMPC (Ultra Mobile PC) [17], with the same detection rate and false alarm rate as on their desktop-based counterpart. Recently we also port the system onto more cost-effective mobile devices, i.e., Windows CE based hand-held devices.

This paper is organized as follows. In section 2 the system architecture and UI features are presented. In Section 3, we describe the robust and efficient language model used in our system. Then in Section 4, we report the experimental results on hand-held devices. Finally in Section 5 we draw conclusions and report latest update of our ongoing work.

2. ART System Architecture

Figure 1 shows the architecture diagram for our system, which includes three layers from bottom to up: Hardware, Operating System, and Application.

2.1. Hardware Layer

This layer describes hardware components on the mobile devices. Fig. 2 shows the two devices used for this study. There are two basic input mechanisms for them: the first one is speech input via a built-in microphone array, and the second one is pointing input via a stylus. UMPC also provides touch input. Each device comes with two USB 2.0 interfaces which enables the use of other peripheral devices such as keyboard, mouse, CD-ROM, etc. And each device has built-in wireless network capability which enables fast data transfer between the devices and backend desktop PC. Each built-in microphone array has two microphone elements. In the future, this setting may be extended to include more elements (4 or 8) to provide better noise cancellation effect. Table 1 gives the system configurations for both devices.

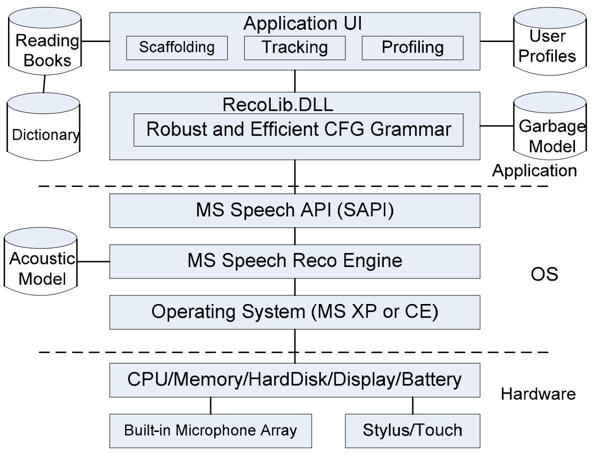


Fig. 1. Architecture diagram of the proposed ART system.

Fig.2. Motion LS 800 Tablet PC (Left) and Samsung Q1 UMPC (Right).

	Motion LS 800 Tablet PC	Samsung Q1 UMPC
CPU	Pentium M 1.2 GHz	Celeron M 900 MHz
Memory	512 MB	512 MB
Display	8.4” SVGA (800×600)	7” WVGA (800×480)
Input	Stylus+Speech	Stylus+Speech+Touch
Battery	3.0 hours	3.5 hours
Dimension	227×170×22(mm)	227.5×139.5×24.5(mm)
Weight	1000 g	779 g

Table 1. Hardware specifications of two hand-held devices.

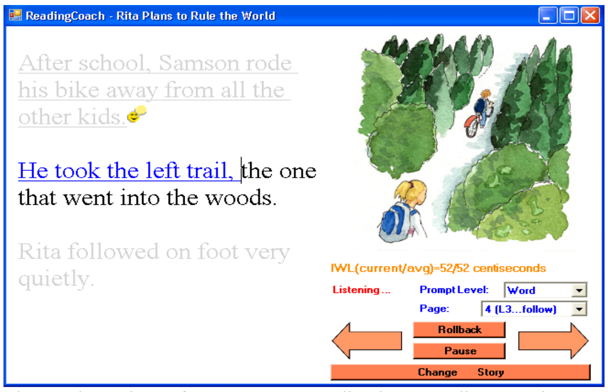
2.2. OS Layer

This layer includes the operating system, speech recognition engine, and Microsoft Speech API (SAPI). The operating system can be Microsoft Windows XP or CE, although here we only tested on XP-based devices. Microsoft Speech Recognition Engine is a state-of-the-art speech decoding engine developed by Microsoft Speech Component Group. It supports both dictation and command-and-control grammars. Currently the engine does not support children’s acoustic model, so we trained this model based on a series of children’s speech data (described in Section 4) with HTK tool and transferred the model format into the one supported by the engine. Microsoft SAPI is one of the standard APIs for speech applications, which has evolved to version 5.3 recently. Both the engine and SAPI are freely available on Windows XP or later version, so we put them on OS layer.

2.3. Application Layer

The Application Layer is composed of two parts. The RecoLib.DLL is an API package we designed for any application using ART features, which in return calls SAPI to do the basic speech recognition functions. One specific feature of this package is that its utilization of our robust and efficient CFG grammar [4], which is built on-the-fly for every sentence or paragraph in reading stories. This grammar can detect most reading miscues when working with our children’s acoustic model, and it reaches real-time tracking speed. This grammar utilizes a garbage model to do the miscue detection. This will be discussed in Section 3.

The other part of Application Layer is the UI components, which include three main functions for tracking, scaffolding, and profiling. The scaffolding UI is based on a “Reading Books” database which includes all the animations, videos, audios and text contents from reading books. The “Dictionary” database is a general knowledge source to provide word-level scaffolding information such as pronunciations, lexicons, grapheme-phoneme mappings, sight words list, or function words list. On the other side, the profiling UI utilizes a database called “User Profiles”. This database is used to store personalization information for each user including reading level, reading progress, a list of hard words, as well as some preference setting.



After school, Samson rode his bike away from all the other kids.

He took the left trail, the one that went into the woods.

Rita followed on foot very quietly.

IWL(current/avg)=52/52 centiseconds

Prompt Level: Word

Page: 4 (L3...follow)

Fig. 3. The main UI for a prototype application “Reading Coach”.

It should be pointed out that the system architecture shown in Figure 1 allows different applications, which may have different UIs but share the same features of ART systems. Fig. 3 shows one of our prototype applications called “Reading Coach”. As can be seen, on the left side, it shows one page of text in the story. The illustration of every page is also shown on the right top side. The right bottom side is the navigation and control panel.

Even in this simple UI, it has included all those aforementioned ART functions:

- A user can select different stories from the reading book database, navigate page-to-page inside the story with or without oral reading/recognition;
- When a user reads the story, it tracks the speech signal against the text. The real-time feedback was provided with different fonts and colors for read and non-read words. As can be seen, only the current sentence is highlighted in black color, and other sentences are in gray color. For current sentence, the underlined font and blue color were used to show the text which has been read and recognized. A cursor is used to prompt the next word to be read.
- A user can click any single word to get the scaffolding help for that word (including grapheme-phoneme mapping and pronunciation). It can also play back pre-recorded speech for a whole sentence (or a paragraph, or a page) by selecting “Prompt Level” at the right-side control panel.

- After finishing reading one sentence, there will be a small "play-me" icon appearing at the end of the sentence so that the user can play back his/her speech recorded by the system by clicking this icon, and the play-back will also show word-level tracking effect.
- A simple profiling function was provided by measuring the reading fluency for each sentence. Here we use Inter-Word-Latency (IWL) proposed by Mostow, et. al. [9] as the fluency metrics. In Fig. 3, it has shown that the IWL for the first sentence is 52 centiseconds (per word), and the average IWL for this page is the same since right now there is only one sentence completed.

In addition, if the system detects any reading miscue (mispronunciation, deletion, insertion, etc.), the cursor will stop right before the first word that the system thinks as mismatched. Different from some other systems, our system does not provide "push" help, i.e., it will not provide help until user asks for it. This may be not the best model, but the benefits are that it leaves more controls to the users and avoids distractions from incorrect "push" helps. Users can easily get help by clicking a word or select to re-play pre-recorded continuous speech. We think this is more convenient especially on mobile devices with stylus.

3. Efficient And Robust Language Model

Children's speech is very different from adults' due to the fact that children's articulation apparatuses and language skills are immature and are dynamically changing with the age [10]. Besides a set of acoustic models trained from a large amount of children's speech data, we are using a dedicated language model in our ART system which is specifically designed for children's speech, in order to detect as many types of reading miscues as possible, including mispronunciations, partial pronunciations, repetitions, filled pauses, as well as other speech and sounds not following the story (e.g., talking to another child, TV sounds, toy making sounds, etc.)[4]. This model is based on one type of "Interpolated Language Model" recently proposed in [11]. It is composed of two paths: a target N-gram path in parallel with a general-purpose garbage model. The target N-gram is built from the current story text, and the garbage model is built from a list of common words in general domain. Also based on the well-known fact that N-gram can be implemented with Context-Free-Grammar (CFG) [12], we are using CFG to represent both paths. The reason of using CFG instead of N-gram is because in SAPI, CFG is much more flexible and can be dynamically changed during runtime so that the grammar can be built quickly with the current story text. Fig. 4 illustrates an example grammar for a given story paragraph or sentence. The target CFG and garbage CFG correspond to the story LM and general-domain LM, respectively. These two paths are connected by a unigram back-off node from the target CFG. <S> and </S> are the entry and exit nodes for the grammar. The two weights shown in Fig. 4 control the possibilities of moving from the target CFG to the backoff node (w_1), and from the backoff node to the garbage CFG (w_2).

Fig. 5 depicts the binary CFG built from a single story sentence "Giants are huge". In addition to the three special states (<S>, </S>, and unigram backoff state), states with one word (e.g., "Giants") are bigram states and states with two words (e.g., "Giants are") are trigram states.

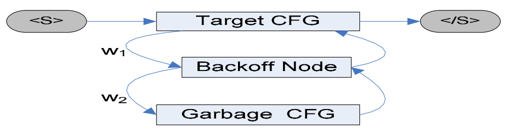


Fig. 4. A schematic illustration for our interpolated LM.

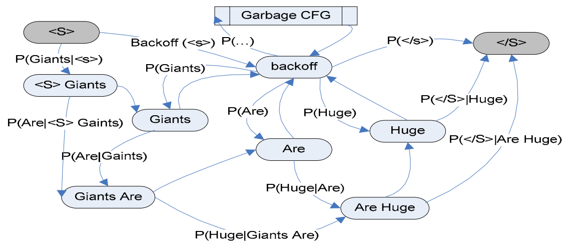


Fig. 5. Interpolated N-gram model for a short sentence ("Giants are huge."). Transitions without labels are back-off transitions.

Our language model is efficient because both target CFG and garbage CFG are compact. The size of the target CFG is small (on the order of only kilo bytes) because the current story text (paragraph or sentence) is short, so the building procedure can be done on-the-fly. The size of the garbage model (usually from 4 KB to 8 MB) can be selected with different levels of complexity from trigram, bigram to unigram. Our experiments showed small differences in detection and false alarm rates using different orders in the garbage model. A further strength of the garbage model is that it can be shared by different sentences or paragraphs which can save loading overhead in runtime. Using this efficient language model is one of the most important reasons to achieve real-time word-level tracking on hand-held devices.

Our language model is also robust in detecting reading miscues because the garbage model is composed of most common words in English so that it can absorb all those words not matched by target model. Even for those out-of-vocabulary words or sub-word level reading miscues, the garbage model can also absorb them by outputting the acoustic-similar words. In [4] we also showed that by using this grammar the system's decision procedure is equivalent to a hypothesis-testing scenario as in the regular utterance verification problem.

Another benefit of using this grammar is the possibility to obtain a ROC curve by adjusting weights (w1, w2) in Figure 4. With this curve, system developers can select best working point of the detection and false alarm rates for different users. Other reported systems usually need post-processing with multiple features of confidence score to obtain such a curve.

4. Experiments

In this section we report testing experiments on two hand-held devices with a published child's reading corpus [2, 13]. The acoustic training data are from four US-English kids' speech corpora including two from University of Colorado-Boulder, one from OGI, and one from CMU. Table 2 lists the speaker number, grades, utterance number, and duration for each part of data used in training and testing.

The test data consist of 105 stories read by 105 children in grades 3, 4, and 5 (17 speakers in grade 3, 28 in grade 4 and 60 in grade 5). There are totally 10 different stories and each story contains an average 1054 words (ranging from 532 to 1926 words).

The children's acoustic model was trained with HTK tools based on EM algorithm which includes 63K Gaussians, and the size of this model is 6132 KB. The language model was built with the method described in Section 3. Since each story was recorded in one wav file, an interpolated CFG grammar was built based on every story with a unigram garbage model (4 KB) including 1600 word in Wall Street Journal domain. The N-gram was built on-line so there is no overhead for language model training.

	Corpus	#Spkr	Grades	#utterance	Time (h)
Train-1	A	665	K~5	39006	26.7
Train-2	B	221	1,2	28829	24.5
Train-3	C	510	1~5	34170	34.5
Train-4	D	76	K~5	5180	9.1
Train-all	ABCD	1472	K~5	107185	94.8
Test	B	105	3~5	105	12.4

Table 2. The training and testing data used in the experiments. A is U. Colorado Kids' Prompt & Read Speech corpus, B is U. Colorado Kid's Read & Summarized Story corpus, C represents OGI Kid's speech corpus, and D represents CMU Kid's speech corpus.

The detection and false alarm rates are computed based on two types of alignments, i.e., one alignment between the story text and human transcription (*story-tra*ns), and the other between the story text and recognition output (*story-hyps*). A reading miscue is defined as an insertion, deletion, or substitution that appears in *story-trans* alignment. If the same reading miscue also appears in the *story-hyps* alignment, it is regarded as a detected miscue. On the other hand, if there is one error in *story-hyps* alignment but no miscue in the same position of *story-trans* alignment, it is regarded as a false alarm. For details of these definitions, see [4,13].

Table 3 gives the results based on the error analyses described above. The detection rate (DT), false alarm rate (FA), and word error rate (WER) on two hand-held devices are shown to be virtually the same as our desktop baseline system with a 3.8 GHz Xeon CPU and 2 GB memory. Similar to [4], online adaptation was not used here (due to multiple speakers) although the engine supports this option. Note that the real-time factor (RTF) reported here does not consider the CPU cycles for UI updating (e.g., cursor moving, font change) since they are difficult to measure for on-line tracking, but this part of overhead in our system is very small. On the whole, our on-line systems have reached the speed faster than real-time on both hand-held devices, although they are much slower than their desktop counterpart. The total memory cost for the prototype system ("Reading Coach") is only 70 MB on both devices.

DT (%)	FA (%)	WER (%)	RTF (xRT)		
			Desktop	Motion	Samsung
71.10	4.40	11.45	0.07	0.17	0.20
72.68	4.92	11.82	0.09	0.21	0.24
75.35	7.30	13.83	0.16	0.38	0.43
75.93	9.01	15.48	0.18	0.46	0.49
76.62	12.01	18.39	0.19	0.47	0.54
76.90	15.80	22.06	0.21	0.48	0.56

Table 3. Experimental results on both desktop and hand-held systems. The weight of garbage model increased from the top row to the bottom row.

5. Conclusion and Latest Update

In this paper we report our initial results in building an ART system that extends children's user experience to hand-held devices equipped with stylus and built-in microphone array. We describe the architecture for our system and its UI components in reading tracking, miscue detection, as well as scaffolding and profiling, as exemplified by one of our applications called "Reading Coach". With the children's acoustic model trained with a large amount of data and an efficient and robust language model, our ART system reaches real-time speed on two small form-factor hand-held devices, with the same detection and false alarm rates as on desktop PCs. Recently we have ported the system onto more cost-effective devices based on Windows CE. A similar ART system has been running on a HTC PDA/Smart Phone with Windows Mobile 6 (400 MHz CPU and 64MB memory). The underlying speech engine is from Microsoft Voice Command 1.6 (plus SAPI 5.0). With a simplified UI, the system also reaches similar accuracy and real-time performance as reported in previous section. The memory usage is only 31 MB and peak CPU cost is 320 MHz. Also we have conducted preliminary usability testing with children on those devices and the results are encouraging. It is noted that the definition of miscue in this paper is stricter than in a real user scenario. For example, if there were two continuous insertions they were counted as two different miscues, but in real scenario the system only needs to report one insertion (or any error) which is enough to catch user's attention. This means the practical detection rate will be much higher than what reported here given the same false alarm rate. Our user study has proved this although the final results need to be measured by pre- and after-use fluency metrics.

6. References

[1] J. Mostow, S. Roth, A. Hauptmann, M. Kane, "A Prototype Reading Coach that Listens", *AAAI*, Seattle, 1994, pp. 785-792.
[2] A. Hagen, B. Pellom, R. Cole, "Children's Speech Recognition with Application to Interactive Books and Tutors," *ASRU*, St. Thomas, USA, 2003.
[3] http://www.soliloquylearning.com
[4] X. Li, Y. C. Ju, L. Deng, A. Acero, "Efficient and Robust Language Modeling for An Automatic Children's Reading Tutor System," *ICASSP*, Honolulu, Hawaii, 2007.
[5] UNSCO EFA Global Monitoring Report, "Strong Foundations: Early Childhood Care and Education," 2007. http://unesdoc.unesco.org/images/0014/001477/147794E.pdf
[6] Report of National Commission for Teaching and America's Future, 2007. http://www.nctaf.org/documents/Unraveling_Shortage_Problem.doc
[7] J. Mostow, "Is ASR accurate enough for automated reading tutors, and how can we tell?" *InterSpeech*, Pittsburgh, 2006.
[8] J. Mostow, G. Aist, et. al., "Evaluation of an Automated Reading Tutor that Listens: Comparison to Human Tutoring and Classroom Instruction," *J. Educational Computing Research*, pp. 61-117, 2003.
[9] J. Mostow, G. Aist, "The Sounds of Silence: Towards Automated Evaluation of Student Learning in a Reading Tutor that Listens," *AAAI*, Providence, RI, pp. 355-361, July, 1997.
[10] S. Lee, A. Potamianos, S. Narayanan, "Acoustics of children's speech: Developmental changes of temporal and spectral parameters," *JASA*, vol. 105, pp. 1455-1468, Mar. 1999.
[11] Y.C. Ju, Y.Y. Wang, A. Acero, "Call Classification by Categorizing Utterances and Using Non-Speech Features," *InterSpeech,* Pittsburgh, PA, 2006, pp.1902-1905.
[12] G. Riccardi, R. Pieraccini, Bocchieri, "Stochastic Automata for Language Modeling," *Computer Speech and Language*, 1996.
[13] K. Lee, A. Hagen, N. Romanyshyn, S. Martin, B. Pellom, "Analysis and Detection of Reading Miscues for Interactive Literacy and Detection of Reading Miscues for Interactive Literacy Tutors," *20th* COLING, Geneva, Switzerland, 2004.
[14] http://www.leapfrog.com
[15] http://www.nintendo.com/channel/DS
[16] http://www.motioncomputing.com/products/tablet_pc_ls.asp
[17] http://samcanhelp.samsungusa.com/samcanhelp/index.jsp