Multi-party Language Interaction in a Fast-paced Game Using Multi-keyword Spotting

Jill Fain Lehman, Nikolas Wolfe and André Pereira

Disney Research, 4720 Forbes Ave, 15213 Pittsburgh, PA, USA.
jill.lehman@disneyresearch.com

Abstract. Existing speech technology tends to be poorly suited for young children at play, both because of their age-specific pronunciation and because they tend to play together, making overlapping speech and side discussions about the play itself ubiquitous. We report the performance of an autonomous, multi-keyword spotter that has been trained and tested on data from a multi-player game designed to focus on these issues. In *Mole Madness*, children laugh, yell, speak at the same time, make side comments and even invent their own forms of keywords to control a virtual on-screen character. Within this challenging language environment, the system achieves 94% overall recall and 85% overall accuracy, providing child-child and child-robot pairs with responsive play in a rapid-paced game.This technology can enable others to create novel multi-party interactions for entertainment where a limited number of keywords has to be recognized.

Keywords: Automatic Speech Recognition, Child-Computer Interaction, Multi-party Interaction, Spoken Dialog Systems

1 INTRODUCTION

Applications using speech recognition are increasingly commonplace, due to both the amount of data available and the maturation of machine learning approaches. Still, existing systems tend to derive their acoustic models largely from adult speech and/or their language models from single-user search and scheduling tasks [2, 8]. As a result, they tend to be poorly suited for young children, whose language is acoustically, lexically, syntactically, semantically, and pragmatically distinct [6, 9]. This is particularly true for young children at play, who are challenging both because of their age-specific pronunciation and because they tend to play together, making overlapping speech and side discussions about the play itself ubiquitous. The same speech recognition issues occur when designing an entertainment application where a child plays a game with an artificial agent or robot that uses natural language for communicating. In this paper, we report the performance of a multi-keyword spotter that has been trained from and tested on data from both child-child and robot-child pairs. The speech recognition system is used to control a virtual on-screen character and it provides children with responsive play in a rapid-paced game, even in the presence of overlapping and out-of-task speech.

2 MOLE MADNESS

Mole Madness is a speech-controlled game in which two players move a virtual mole through its environment, acquiring rewards and avoiding obstacles (Figure 1, left). One player creates horizontal movement using the word *go* and the other creates vertical movement with *jump*, a simple design that makes the game accessible to even the youngest child with little instruction. Successful play requires both coordinated turn-taking and overlapping speech [7].

The game can be played in a child-child or child-robot configuration with Sammy, a Furhat robot head [1] encased in a cardboard body (Figure 1, right). The architecture that permits this flexibility includes separate processes for the game engine, speech recognition, and robot control, coordinated by an IrisTK dialog module [10] that synchronizes the independent processes at time-slice boundaries. The mole character and basic game play are built in Unity3D and include the A* search that returns a value when Sammy requests a move on the optimal path. If the returned value is *go* or *jump*, Sammy randomly plays a pre-recorded sound file with one or more instances of that command. The frenetic pace of the game makes Sammy engage in overlapping speech and the pre-recorded sound files include all the language phenomena identified in the next section.

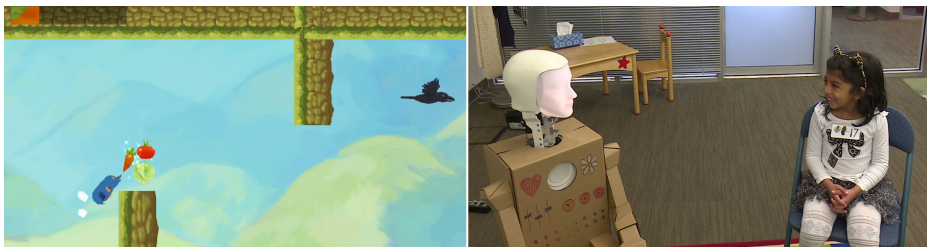

Fig. 1. A Mole Madness screenshot (left) and a play session with Sammy (right).

3 THE SPEECH CORPUS

Data for the models was collected from 62 children between the ages of five and ten who played *Mole Madness* as part of a multi-activity study in summer 2015. Participants' mean age was 7.45 years (SD=1.44 years), and 48% were female. Children were compensated for their participation.

Interleaved with other activities, children played the game twice, first paired with another child and then with Sammy. Each game traversed four to six levels, depending on time available and the child's desire to continue. Speech recognition was performed by a human wizard who listened via headphones in a separate room, trying to map each *go* and *jump* into a button press on a game controller.

To create the ground truth for training the keyword models described below, ˜6.9 hours of gameplay were hand-segmented and transcribed, producing ˜11.8K non-overlapping instances of *go*, ˜9.4K non-overlapping instances of *jump*, ˜10.1k instances of overlapping keywords, ˜2.1K social utterances, and ˜12.9k background segments. The resulting corpus contains three language phenomena that define the main challenges for speech recognition and full autonomy in the mole's behavior:

Overlapping speech: *Mole Madness* was specifically designed to elicit overlapping keywords at a small number of predictable obstacles on each level. However, most children discovered that overlapping speech makes the mole "fly" over flat ground. As a result, almost 40% of keywords overlap in child-child games, and 26% in child-robot games.

Social side talk: When children play together, task commands (*go*, *jump*) naturally occur intermixed with both non-task speech directed to the mole ("watch out," "faster") and speech directed to the other player ("a giant tomato," "he's funny"). Any speech that is not a keyword becomes a potential source of false keyword recognitions. Meta-comments about game strategy ("don't say jump yet") are particularly challenging because they can contain one or more task words that should not be interpreted as commands. On average, 7% of utterances during gameplay were social in nature, and 29%/7% of child-child/child-robot side talk contained at least one *go* or *jump*.

Lexical variability: A game with only two commands is easy to learn but ultimately frustrating in its lack of expressive power. Children always began by imitating the fully-formed versions of the keywords modeled in a brief tutorial video. However, children throughout our age range eventually tried to increase the task vocabulary ("double jump," "go faster"). When that failed, they created variations of the keywords using elision, repetition, and elongation ("g- g- g- g- go," "juuuuuuuuuump") to encode more complex meanings. Based on the distribution of keyword lengths, we define *fast* and *slow* speech to correspond to a keyword with a length that is less/more than half a standard deviation from the mean length, respectively. We posit that when the child uses a typical *go* or *jump* s/he expects to see an instance of the action per word within a causally-meaningful period of saying it. We interpret the meaning of fast speech relative to that norm - faster speech intends faster movement. Slow speech, conversely, appears to have two distinct meanings. Emphatic elongation ("gooo!") seems to ask for a single bigger movement or a movement right away, while prolonged elongation seems to ask for steady or on-going movement. Children used faster-than-normal forms about 32% of the time with each other and the robot but were much more likely to use slower-than-normal speech with the robot (27% of keywords) than the other child (17%).

4 THE MULTI-KEYWORD RECOGNIZER

With only two in-task words and the necessity of reacting quickly enough to establish a perceived relationship between the spoken command and the mole's

action, the recognition problem in *Mole Madness* is a natural fit to keyword spotting in continuous speech. In this section, we describe the components of the real-time implementation of our multi-keyword recognition algorithm [11], including extensions to that work necessitated by the language phenomena outlined above. Figure 2 grounds the discussion.

In most example-based keyword spotters, training data is used to build a model of the keyword in its entirety and a window on the speech stream that is the size of the expected duration of the word is evaluated against the model in a sliding fashion. As more of the keyword appears in the window, the probability increases that the necessary threshold to signal recognition will be reached. Previous work [11] extended this idea by building separate models of both non-overlapping and overlapping speech, then viewing the speech stream in the window as composed from a probabilistic mixture of those models, using Student's t-distribution (hereafter, TMMs). Models in that work were based on a 300ms window - the mean duration of *go* and *jump* in the data. The evaluation of the algorithm used pre-segmented keywords, with a categorization of the segment as a whole into one of the classes *go*, *jump*, or *mixed* depending on the most prevalent classification as the window slid across the entire segment.

The online version also models overlap explicitly but extends the previous work in a number of ways. The TMMs in Figure 2 are trained using the same algorithm and hand-annotated corpus, but with a 150ms window size and an extended set of classes: *go*, *jump*, their combination (*mixed*), *social speech* and *background* noise segments. The recognition system as a whole can issue at most one command to the mole per time-slice, indicating if *go*, *jump* or both were spoken. The shorter time-slice allows faster response time overall as well as better recognition of individual elided forms in fast speech ("ju- ju- j- jump"). The latter means the mole will better conform to the expectation that faster speech leads to faster movement, but a shorter time-slice also means that less context is available for distinguishing between the keywords themselves and discriminating task speech from social speech. The addition of a distinct *social speech* classifier is used in conjunction with other compensatory features of the system to address this problem, as described below. Similarly, the addition of an explicit *background* model is included to control for another source of false keyword recognition.

At run-time, we perform the maximum-likelihood computation as shown in the middle portion of Figure 2. The algorithm extracts overlapping blocks of Mel-frequency cepstral vectors (MFCCs,) within the 150ms time-slice, and then the TMMs are used to compute the posterior probabilities for each of the five class labels with the WEKA library.

The simplest way to translate the output of the TMM classifiers into game commands would be to define a constant posterior threshold, as in [11] and many other systems. Under such an approach, a *go*, *jump* or *mixed* command would be sent to the game at the end of the time-slice if the posterior exceeded the respective classifier's threshold, and no game command would be sent if all of those posteriors were below the threshold. Given atypical lexical forms and the intermix of keywords and social speech, we need and can get more power by

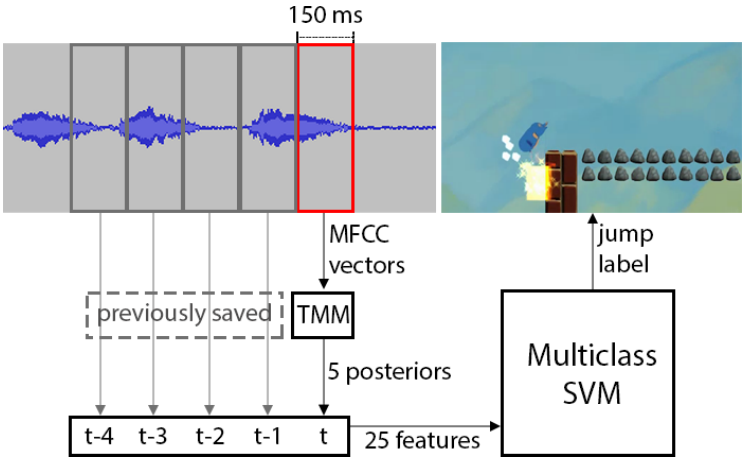

Fig. 2. A sample classification task: audio features of the current 150ms time-slice are combined with information from four previous time-slices to generate a *jump* label.

looking at the relative likelihoods of each class in combination and over time. As shown in Figure 2, we achieve a finer-grained judgment based on the patterns of probabilities in the training data by building an additional classifier over the combination of posterior values from the TMMs for the current time-slice and several previous time-slices. We tested several algorithms - decision trees, neural networks and Support Vector Machines (SVMs) - and several values for the number of prior segments. Best performance was achieved with four prior segments (600ms of history) and a multi-class SVM model with a Radial Basis Function (RBF) kernel using the open source LibSVM library and performing a grid search to find the optimal C and gamma values.

The use of prior history is intended to overcome the information that was lost by committing to a more reactive 150ms time-slice. The solution as a whole - using the full context of the relative likelihood of the five different classes of speech over time - is intended to help distinguish keywords embedded in social speech from keywords that are spoken as task commands. It should also help to discriminate partial keywords that occur at the time-slice level during both elongation and elision from those same sounds when they occur as components of non-task utterances.

5 METHOD OF EVALUATION

A metric of evaluation must take into account the contextual and temporal aspects of the real-time game environment. Although the system makes a decision at the end of every 150ms, accuracy statistics at this architectural level are misleading for two reasons. First, time-slice level accuracy gives undue importance to correct recognition behavior during silence, which constitutes 45% of any game, on average. Indeed, children that were shy, slower to learn the game, or

not fully engaged would have high accuracy measures even in cases where the recognizer was doing poorly on the spoken phenomena required to play.

Second, and more generally, the time-slice is not the unit of measure at which the phenomena of interest occur. These phenomena include not only the linguistic ones - variable-length commands, overlapping speech, and social side talk - but also, critically, the child's perception of the causal relationship between her/his words and the mole's action. The purpose of evaluation is to establish whether the word spotter creates an interaction that corresponds to the child's natural understanding of cause and effect: if one or both keywords are spoken to the mole, the relevant action(s) should be perceived to occur in response; if a keyword is spoken incidentally in a social context, or if no keyword has been spoken, then no corresponding change in the mole's behavior should be seen.

To bridge the divide between recognizer output at the 150ms time-slice and human judgments at the level of variable-length but semantically-meaningful units, we aggregate the behavior of the recognizer across time with respect to the annotation. The window over which we aggregate reflects assumptions about how long a lag there can be between voicing the command and seeing the mole's behavior change before the child no longer experiences the two as causally connected. Choosing the appropriate recognition window is not a trivial task because it depends on the particular activity or game [3]. The wizard who mapped spoken commands to button presses in our corpus, for example, had a mean reaction time of 529ms (SD=419ms). Despite the variable and occasionally significant lags in the wizard's response, children appeared to experience the game as volitional (if a bit finicky) and enjoyed it overall [11]. These results are in line with both [5], which found a 300-700ms lag to be acceptable, and [4], which found that a 400ms lag produced a greater sense of agency than an 800ms lag in a task where two people were engaged in potentially overlapping behaviors.

Because our application is a fast-paced, real-time game and children's perceptual expectations may be variable, we compute performance statistics both with and without a lag. In the no-lag case, the window over which recognizer output is aggregated extends from the beginning of the annotation through the time-slice in which the annotation ends. This method compensates for imprecision in the annotator's boundaries, holds the system to a tight standard for causality, and defines a lower-bound on performance from a perceptual perspective. In the with-lag case, we follow the literature, allowing for as much as 450ms between the end of speech and action. The longer window extends from the beginning of the annotation through three time-slices after the end of the annotation. It may give credit for detecting a keyword based on evidence that falls outside the shorter window's view, but does so under the assumption that children would also attribute the mole's action to their utterance within that period.

To compute standard statistics for the multi-keyword recognizer, each annotation in the corpus and each continuous segment of silence must be accounted for. A ground truth keyword annotation counts as either a **false negative** (FN) or a **true positive** (TP). A false negative is scored when there is no time-slice with the keyword's label in the window. Similarly, a true positive is scored if

there is at least one time-slice with a matching game command in the window. Thus FN means the character's movement doesn't change as a result of the spoken word and TP means that it appears to do so. A time-slice labeled as *mixed* represents both a *jump* and a *go* command for evaluation purposes.

The TP definition leads to a number of consequences. It means, for example, that a single keyword annotation in the corpus "consumes" all the matching detections within the window's bounds. In the case of average length keywords this is likely to correspond to one, occasionally two, commands per annotation. In the case of fast speech, it means that the system's performance is bounded by the time-slice - the recognizer can get credit for an elided form, at most once every 150ms. If the system is working well, it will nevertheless recognize enough of those elided forms that the mole's behavior will reflect the child's intent: faster speech will create faster movement. For a slow keyword, however, the TP definition biases the statistics in the recognizer's favor, potentially giving full credit to a five second *go* that has only an occasional correctly-labeled time-slice in it even though the apparent behavior would not correspond to the steady movement that is expected. To remove this bias, we preprocess slow speech into separate consecutive 300ms keyword (normal duration, steady movement) segments and apply the TP/FN definitions to each segment individually.

The remaining annotations - social speech and, by default, non-speech segments with silence and/or background noises - are the potential sources of *true negative* (TN) judgments. Both types are scored as TN if there is no time-slice with a keyword label in the window. Note that this way of treating silence minimizes the influence of TN on accuracy in the same way that an architectural time-slice accounting would maximize its influence.

Social speech and non-speech segments can also be sources of **false positives** (FP). An FP is scored when a keyword label generated by the recognizer does not fall within the window of any keyword annotation. A false positive is also scored for any isolated (non-overlapping) keyword that is recognized as an instance of the other keyword.

6 RESULTS AND DISCUSSION

Table 1 summarizes the results of a 10-fold cross-validation on the corpus, calculated with and without lag. Overall the system's accuracy is 85% with the more conservative metric, and 89% with the wider perceptual window. As expected, the increase in accuracy is attributable primarily to keywords that are recognized during time-slices after the end of the annotation.

Note that the poor specificity (with high variability) is attributable largely to our conservative method of scoring silence and social talk. Because such segments are counted as a single unit, a false positive means only that there was at least one time-slice labeled with a keyword during the span. For elongated keywords, we divided the annotation into contiguous 300ms segments to be able to detect whether the recognizer would produce the expected continuous movement. For stretches of silence and social talk, however, we have no *a priori* understanding

Table 1. Performance means (standard deviations) with and without perceptual lag, overall and separated by co-player type. Precision is the fraction of correctly identified positive results (TP/(TP+FP)). Specificity is the fraction of correctly identified negatives (TN/(TN+FP)). Sensitivity is the proportion of positives that are correctly identified as such (TP/(TP+FN)).

	No Lag			With Perceptual Lag		
Session Type	Overall	Child-Child	Child-Robot	Overall	Child-Child	Child-Robot
Accuracy	.85 (.10)	.79 (.11)	.89 (.07)	.89 (.09)	.83 (.10)	.93 (.06)
Precision	.85 (.10)	.78 (.11)	.89 (.07)	.90 (.09)	.83 (.10)	.93 (.06)
Specificity	.70 (.19)	.57 (.20)	.76 (.14)	.77 (.18)	.64 (.18)	.84 (.13)
Sensitivity	.94 (.08)	.92 (.10)	.96 (.06)	.96 (.07)	.93 (.09)	.97 (.05)
Go	.95 (.08)	.93 (.11)	.96 (.05)	.96 (.07)	.94 (.09)	.97 (.04)
Jump	.94 (.09)	.91 (.10)	.96 (.08)	.95 (.08)	.92 (.09)	.97 (.08)
Overlapping	.93 (.09)	.91 (.11)	.95 (.07)	.95 (.08)	.93 (.10)	.96 (.06)
Slow	.95 (.08)	.92 (.08)	.96 (.07)	.96 (.06)	.94 (.08)	.97 (.05)
Medium	.95 (.07)	.92 (.10)	.97 (.04)	.96 (.06)	.94 (.08)	.98 (.03)
Fast	.92 (.12)	.85 (.15)	.95 (.08)	.94 (.11)	.88 (.13)	.97 (.08)

about how often a sporadic unexpected movement can occur before the causal connection between speech and action is shattered. Were we to break background and social segments up using the same 300ms rule, the specificity would increase to 84%/88% in the no lag and perceptual lag conditions, respectively. Nevertheless, the values under the stricter accounting shown here are important for giving a realistic idea of how well the system handles keywords in side talk. The results are less favorable than we would like: social speech with one or more embedded task words still triggers a misrecognition about 65% of the time, despite the information in the four prior time-slices. The fact that social segments without embedded task words also generate a misrecognition about half the time suggests that we may have erred too much on the side of responsiveness in choosing the time-slice. A somewhat longer time-slice might change the number of errors that come from false recognition of sub-segments of non-task words without substantively affecting the rest of the children's experience. Alternatively, achieving a better balance in the amount of keyword versus social speech data might also help minimize this source of error.

The remaining source of misrecognition (FPs) is the confusion of one keyword for another. This confusion occurs almost entirely from overprediction of the *mixed* category in the presence of a single keyword. The problem seems to stem from a strong correlation between overlapping keywords and volume - when they are excited both children are more likely to be yelling and commanding. As a result, even when children yell a single command alone, the combination serves as evidence for the *mixed* model.

Overall sensitivity is excellent across all conditions, although statistically better and less variable during child-robot games. Higher values occur with the robot co-player because the robot's voice is less varied and more predictable. We

intentionally treat the robot like a player with a voice that must be recognized, despite the fact that it is possible to know with certainty when the robot issues a command. Having a uniform solution for both types of co-player simplifies the architecture and allows any agent with natural language synthesis to play our game. As important, it creates a grounded experience for the child given that some errors in detection and the same lag in response can be noticed.

Looking at sensitivity more closely, we find that it is consistent across keywords and not statistically different for non-overlapping and overlapping speech. The one class of phenomena in which we do see differences in recognizer performance is lexical variability: performance is statistically worse on fast speech when compared to either slow speech or typical duration forms. As noted above, false negatives must occur whenever the keyword rate is faster than the 150ms time-slice. In addition, shortened forms contain less signal and are more error-prone in general. Runs of fast speech occur more often in child-child games, and when they do occur in child-robot games they do so in Sammy's more easily recognized speech about 40% of the time. As a result, the degradation in performance is significant only in child pairs. Note that although sensitivity is about 7% worse in fast speech at 150ms, it is, nonetheless, recognized at least 85% of the time, often enough to ensure that rapidly repeated keywords produce perceptually faster movement in the mole.

To set these results in a larger context, we compared the performance of the multi-keyword spotter without perceptual lag to the performance of a state-of-the-art commercial continuous speech recognizer given the finite state grammar *go* and *jump*. The commercial system was about 35% less accurate overall, and displayed a number of biases not seen in our results. In particular, where the word spotter detects *go* and *jump* about equally well, the commercial recognizer was significantly worse at detecting *go* (67% sensitivity for *jump* versus 40% sensitivity for *go*). Similarly, the commercial system did more poorly on overlapping than non-overlapping speech (25% versus 46%) and showed uniformly decreasing performance as a function of keyword rate (50% sensitivity on slow words, 45% on normal duration, and 19% on fast speech).

7 CONCLUSIONS AND FUTURE WORK

The motivation for this work was to explore a point in the space of multi-party language-based character interactions for young children that confronted head on some of the difficult issues that arise with children at play. *Mole Madness* allows us to study overlapping speech, side talk, and exaggerated variability in pronunciation. Within this challenging environment, the system described in this paper is able to achieve 94% overall sensitivity and 85% overall accuracy. The technology presented here can be reproduced with other vocabulary, allowing designers and developers to build novel children's applications that use limited speech to the agent as an input method.

Although our solution works well across most of the phenomena, and significantly better than commercial systems, a number of important questions

remain. The first question concerns the generality of the results. *Mole Madness* was designed for only two players and with only two keywords. Although we can imagine many scenarios with most or all of these same characteristics (two children being asked by a character if they want to go left or right, for example), it is important to understand how performance systematically degrades as a function of relaxing these assumptions. What happens when three or more voices are calling out commands? Which results change when the keywords are of varying length, or have common phones or syllables? And, of course, how many keywords can be adequately distinguished given any one or more of these modifications?

References

1. Al Moubayed, S., Beskow, J., Skantze, G., Granström, B.: Furhat: A Back-Projected Human-Like robot head for multiparty Human-Machine interaction. In: Cognitive Behavioural Systems, pp. 114–130. Lecture Notes in Computer Science, Springer Berlin Heidelberg (2012)
2. Bellegarda, J.R.: Spoken language understanding for natural interaction: The siri experience. In: Natural Interaction with Robots, Knowbots and Smartphones. pp. 3–14. Springer New York (2014)
3. Claypool, M., Claypool, K.: Latency and player actions in online games. Commun. ACM 49(11), 40–45 (Nov 2006)
4. Dewey, J.A., Carr, T.H.: When dyads act in parallel, a sense of agency for the auditory consequences depends on the order of the actions. Conscious. Cogn. 22(1), 155–166 (Mar 2013)
5. Edlund, J., Edelstam, F., Gustafson, J.: Human pause and resume behaviours for unobtrusive humanlike in-car spoken dialogue systems. EACL 2014 p. 73 (2014)
6. Gerosa, M., Giuliani, D., Narayanan, S., Potamianos, A.: A review of ASR technologies for children's speech. In: Proceedings of the 2Nd Workshop on Child, Computer and Interaction. pp. 7:1–7:8. WOCCI '09, ACM, New York, NY, USA (2009)
7. Lehman, J., Al Moubayed, S.: Mole madness–a multi-child, fast-paced, speech-controlled game. In: AAAI Symposium on Turn-taking and Coordination in Human-Machine Interaction. Stanford, CA (2015)
8. Liao, H., Pundak, G., Siohan, O., Carroll, M.K., Coccaro, N., Jiang, Q.M., Sainath, T.N., Senior, A., Beaufays, F., Bacchiani, M.: Large vocabulary automatic speech recognition for children. In: Sixteenth Annual Conference of the International Speech Communication Association. research.google.com (2015)
9. Shivakumar, P.G., Potamianos, A., Lee, S., Narayanan, S.: Improving Speech Recognition for Children using Acoustic Adaptation and Pronunciation Modeling. Proceedings of Workshop on Child Computer Interaction (Sep 2014)
10. Skantze, G., Al Moubayed, S.: IrisTK: A statechart-based toolkit for multi-party face-to-face interaction. In: Proceedings of the 14th ACM International Conference on Multimodal Interaction. pp. 69–76. ICMI '12, ACM, New York, NY, USA (2012)
11. Sundar, H., Lehman, J.F., Singh, R.: Keyword spotting in Multi-Player voice driven games for children. In: Sixteenth Annual Conference of the International Speech Communication Association (2015)