Midterm: Improving Responsivity of the InMind Movie Agent Through Incremental Processing

Vivian Tsai

Abstract

Carnege Mellon University's InMind team is currently working on an intelligent personal assistant who, through a mobile application, connects socially with its user while completing a specific task. The following paper details the current latencies within this dialog agent's architecture, as well as an implemented solution, involving incremental processing, to increase its response time and the effects of this solution on agent-user interaction.

1 Introduction

The InMind Movie Agent, developed as part of Carnegie Mellon's collaborative InMind project, uses genre, director, and actor preferences to recommend movies to a user through a mobile application setting. We define a **turn exchange** for the agent as a unit that begins when the user presses the app button to speak and ends when the movie agent initiates a reply; one turn exchange thus encompasses the user's speech, the processing of that speech and generation of a response, and the agent's output of said response. A conversation between a user and the movie agent will consist of several turn exchanges.

At present, there exists a significant delay, comprised of multiple latencies from different architectural components, within each turn exchange–specifically between the end of a user's utterance and the start of the dialog agent's ensuing response. Our goal is thus to diminish the overall delay by identifying and eliminating the largest contributors within this time span.

Possible relevant solutions for eliminating latencies include **incremental processing** regarding the dialog agent's response and **speculative execution** regarding partial ASR results. Past research indicates that incremental systems are preferred by users and rated not only as faster, but also as more efficient and polite.

In this paper, we examine the results of an in-depth analysis of latencies within turn exchanges; use those results to determine which of the mentioned solutions would prove effective if implemented; discuss the strategies used to implement these solutions; and note the impact of these implementations through the results of a comparative user study. Specifically, we study how the elimination of identified time delays through an incremental processing approach alters interactions between agent and user, as well as the user's evaluation of such interactions.

2 Analysis of Latencies

2.1 Overview

We can categorize turn exchanges into two groups: those which involve queries to the movie database, and those which do not. The following data are compiled from 115 turn exchanges, where 42 fall in the former category; these exchanges were obtained from conversations between ArticuLab members and the dialog agent.

Time delays within these turn exchanges were identifed through the use of an analysis tool, which was developed to parse conversations, distinguish unique turn exchanges, and collect various statistics, the most significant of which are detailed in the following section.

2.2 Analysis Tool

The analysis tool utilizes output from log files of the multiuser framework (MUF) server, phone client, and NLU/DM, using the timestamps of log messages to calculate the time spans of various processes.

2.2.1 Structure

Within the analysis tool, the *TurnExchange* object, representing a turn exchange, is comprised of an enum map; each key is a unique *EntryType*, and each value is the *LogLine* object (containing data from a specific log line) that corresponds to that key.

An *EntryType* (enum type) represents a unique action (corresponding to a log message) that takes place during a turn exchange (i.e. the social reasoner (SR) obtaining a strategy to implement). The complete set of EntryType variables, as well as their chronological ordering within a turn exchange, is detailed in Appendix A.

2.2.2 Calculations

The following statistics are collected by the analysis tool (and discussed in the Results section).

Timing for Modular Components As the log messages within logs often indicate the start or end of processes, the difference between the timestamps of two specific log messages translates to the duration of the specific modular process that they encompass. The TurnExchange function

$$duration(entryType1, entryType2)$$

gives the time in milliseconds between the occurrences of $entryType1$ and $entryType2$, respectively. When applied to several TurnExchange objects, the function ultimately provides a dataset of latencies (for a particular module) that we can evaluate.

Timing for Speech Results For each TurnExchange object, the incremental speech results of Google ASR are collected chronologically and stored, with their corresponding timestamps, in a list. This allows for comparison of partial and full results, as well as time delays between a specific speech result and the subsequent final result.

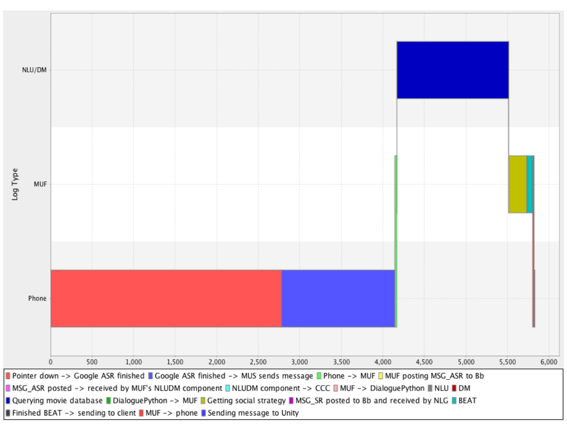

Figure 1: Timeline overview of TurnExchanges with queries

2.3 Results

2.3.1 Component Latencies

A timeline overview can be seen in Figure 1. The resulting statistics show four significant latencies, where a "significant latency" is defined as one with a duration over .1000s:

Pointer down to Google ASR output This latency can be ignored, as it measures the delay between the user's pressing of the app button and the Google ASR's output of that user's utterance. This has no correlation with the length of the user's utterances or duration for which s/he speaks; moreover, this latency is dependent on the user's actions only and is not a result of the InMind architecture.

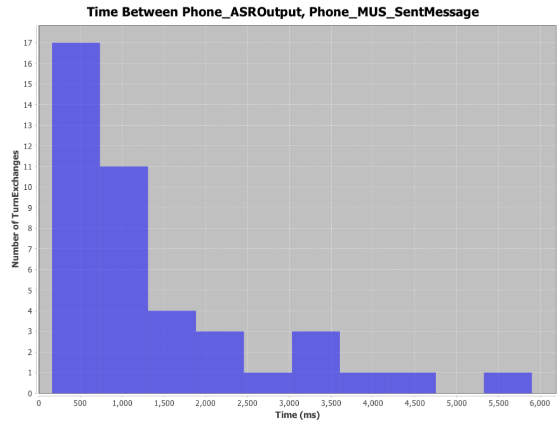

Figure 2: Google endpointing delay

Google endpointing The time delay between Google ASR's return of a final output and the sending of that output to the multiuser framework. These results are displayed in histogram form in Figure 2.

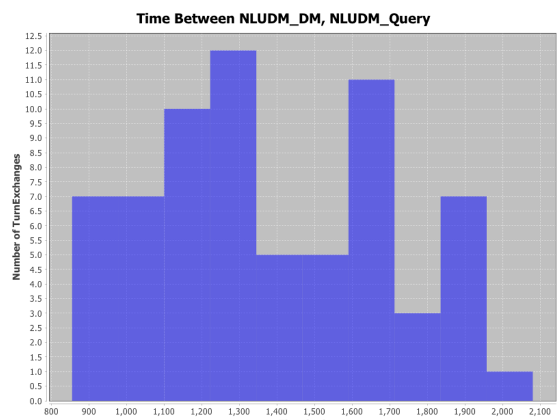

Figure 3: Query time for TurnExchanges with queries

NLU/DM Query A movie database query takes a mean of 1.342s and a median of 1.267s, with a standard deviation of .2918s. These results are displayed in histogram form in Figure 3.

Getting social strategy The social reasoner currently takes a mean of .2164s and a median of .2850s, with a standard deviation of .1105s. However, as this latency is quite small compared to that of the ASR output and that of the movie database query, there is not an immediate need to make changes. We propose to re-evaluate this time delay once the larger ones have been [fixed].

2.3.2 Speech Service Latencies

Our analyses from the previous section do not provide enough evidence that implementing speculative execution would be beneficial. Based on incremental speech results from 114 turn exchanges, the average delay between the [speech service's generation of the correct final result] and the ASR's finalization of that result is .4876s, with a standard deviation of .5818s; note that this yields a coefficient of variation of over 1, and we thus conclude that there is no consistent delay that could be approached.

Futhermore, Appendix B.2 shows that the delay between realization of a meaningful, partial ASR result and realization of the complete ASR result is neither constant nor significant Out of 55 utterances, only 36 had partial results (that leaves 34.55% without partial results).

2.4 Proposed Solutions

Based on the aforementioned results, we propose the following actions in order to reduce time delays and thus improve functionality:

- **Investigate middleware to find cause of ASR output delay** We propose a closer look at the middleware to better understand the time delay between the issue of the user's utterance and the sending of that utterance to the multiuser framework, particularly since no action appears to be taken within this time delay.

- **Introduce incremental results within the InMind architecture** Rather than work to reduce the duration of a movie database query, we seek to *fold* this latency into the InMind movie agent's social sentence through incorporating incrementality. This proposal (including the concept of *folding*) will be further explored in the next section.

We additionally conclude that incorporation of speculative NLU will not prove beneficial at this time and will thus not be implemented.

3 Incremental Processing

3.1 Folding Query Time

In terms of the InMind movie agent, a *social sentence*, underlined in the examples below, refers to the utterance preceding the agent's actual question or movie recommendation to the user:

> I like the way you think! Who are your favorite directors?

> Wow, here is one I'd love to go to. It's called *Oblivion* (2013).

> I think this movie fits your tastes. How about *Edge of Tomorrow* (2014)?

The shortest social sentence (in terms of both audio and word count) that the InMind movie agent produces–"I think this movie fits your tastes"–takes approximately 1712ms (1.712s) for the agent to speak. This is more than enough time for the 1.342s movie query time to be *folded* into the social sentence, where *folding* here means having the InMind agent verbalize the social sentence while the movie database is being queried; recall that the query result is only required for the agent's actual recommendation (second sentence).

Note that for responses without a social sentence, which are currently of the form "How about [movie title]?" and only occur when no social strategy is set, adding a simple phrase like "Let me think for a moment", which takes a mean of 1.342s, will still add sufficient time for folding.

3.2 Overview of Changes

At present, the dialogue manager (DM) handles queries by outputting a Recommendation object–which includes the movie to be recommended–once the movie database has been queried. The social reasoner then selects a strategy; the NLG generates an appropriate response (consisting of a social sentence and a movie recommendation sentence), filling in the latter with the movie title from the Recommendation object; and the NLG response is sent to speech. This flow of events is represented in Figure 4 below:

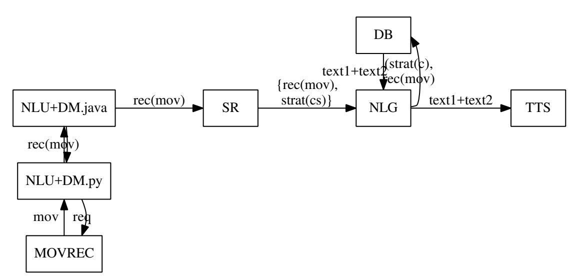

Figure 4: Current system

We instead propose an incremental approach (Figure 5):

1. The DM outputs the Recommendation object, which contains an underspecified variable in lieu of an actual movie title.

2. While the DM queries the movie database for a movie title, the social reasoner selects a strategy, and the NLG generates an appropriate response,

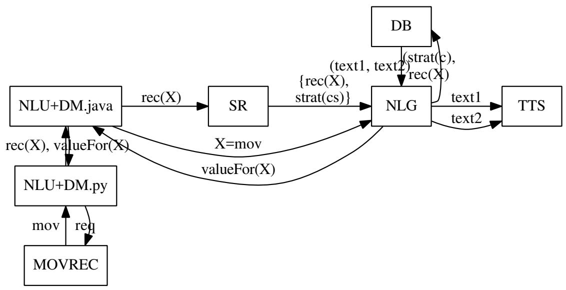

Figure 5: Proposed system

with the social sentence and movie recommendation sentence as two separate entities.

3. As the social sentence contains no variable, the NLG immediately forwards it to speech.

4. While the social sentence is being spoken by the InMind movie agent, the DM finishes its movie query and sends the result to the multiuser framework; as it is now able to retrieve a value from the Recommendation object, the NLG replaces the variable in the movie recommendation sentence with the newly given movie title. The movie recommendation sentence is then forwarded to speech.

5. Since a movie query takes less time than the verbalization of a social sentence, TTS will wait until the social sentence is finished and then output the movie recommendation sentence.

3.3 Modular Changes

To implement this incrementality, the following components of the InMind system will require modifications:

NLG Component The NLG component will need to process the social sentence and recommendation sentence of an intent separately. The social sentence can be processed immediately; the recommendation sentence, containing the variable of a recommended movie title, must await further information from the NLU/DM.

DM on Python side If a movie recommendation is necessary, the DM will need to handle two queries, the first being the underspecified variable and the second being the actual movie result.

Text to speech To prevent the InMind movie agent from interrupting itself once the recommendation sentence arrives, TTS needs to be aware of when the agent has finished speaking and, meanwhile, add the received recommendation sentence to a queue.

A EntryTypes

A.1 List of EntryTypes

An EntryType represents a unique log message found within a TurnExchange. Each EntryType name roughly corresponds to LogType_Component_Action.

Phone_TTS_Initialized
Phone_Unity_PointerDown
Phone_Unity_ServerStarted
Phone_SpeechService_GettingOutput
Phone_Unity_PointerUp
Phone_ASROutput
Phone_CCC_Message
Phone_MUS_SentMessage
Phone_SpeechService_APIComplete
MUF_Orchestrator_ReceivedMessage
MUF_Orchestrator_UtteranceFromASR
MUF_RE_ReceivedBlackboard
MUF_RE_ReceivedMessage
MUF_NLUDM_ReceivedMessage
MUF_NLUDM_SentGreeting
MUF_NLUDM_ToBlackboard
NLUDM_ASR
NLUDM_NLU
NLUDM_DM
NLUDM_Query
MUF_NLUDM_ReceivedAction
MUF_SR_SetStrategy
MUF_NLG_ReceivedStrategy
MUF_NLG_ToBlackboard
MUF_Orchestrator_ToClient
MUF_NLG_BSONToAndroid
MUF_NLG_Sentence
Phone_CCC_Request
Phone_MultiuserEvent_Request
Phone_MessageToUnity
Phone_Unity_MessageFromAndroid

A.2 EntryType Timelines

Tables 1 and 2 below show the chronology of log messages for initial turn exchanges and subsequent turn exchanges, respectively.

Table 1: Chronological timeline of log messages, first exchange

Phone log	MUF log	MUF/DM log
TTS_Initialized		
Unity_ServerStarted		
	Orchestrator_ReceivedMessage	
	MUF_Orchestrator_UtteranceFromASR	
	NLUDM_ReceivedMessage	
	NLUDM_ToBlackboard	
	NLUDM_SentGreeting	
		ASR
		NLU
		DM
		Query
	NLUDM_ReceivedAction	
	SR_SetStrategy	
	NLG_ReceivedStrategy	
	NLG_ToBlackboard	
	Orchestrator_ToClient	
	NLG_BSONToAndroid	
	NLG_Sentence	
CCC_Request		
MultiuserEvent_Request		
MessageToUnity		

Table 2: Chronological timeline of log messages, subsequent exchanges

Phone log	MUF log	MUF/DM log
Unity_PointerDown		
SpeechService_GettingOutput		
Unity_PointerUp		
ASROutput		
CCC_Message		
MUS_SentMessage		
	Orchestrator_ReceivedMessage	
	MUF_Orchestrator_UtteranceFromASR	
	RE_ReceivedBlackboard	
	RE_ReceivedMessage	
	NLUDM_ReceivedMessage	
	NLUDM_ToBlackboard	
		ASR
		NLU
		DM
		Query
	NLUDM_ReceivedAction	
	SR_SetStrategy	
	NLG_ReceivedStrategy	
	NLG_ToBlackboard	
	Orchestrator_ToClient	
	NLG_BSONToAndroid	
	NLG_Sentence	
CCC_Request		
MultiuserEvent_Request		
MessageToUnity		

A.3 EntryTypePairs

The following section lists the EntryTypePairs currently considered by the analysis tool (for pie chart and timeline graphics), along with explanations of the processes each EntryTypePair represents.

Phone_Unity_PointerDown, Phone_Unity_PointerUp The duration for which the user holds down the button in the InMind movie agent app. It is important to note that this duration does not correlate to the user's utterance in any way and is not a result of the InMind architecture.

Phone_Unity_PointerUp, Phone_ASROutput The time between the user's release of the app button and Google ASR's output of the user's utterance.

Phone_ASROutput, Phone_MUS_SentMessage The time between Google ASR's output and the multiuser service's sending of that output.

Phone_MUS_SentMessage, MUF_Orchestrator_ReceivedMessage Time it takes for the message to travel from phone to multiuser framework.

MUF_Orchestrator_ReceivedMessage, MUF_Orchestrator_UtteranceFromASR Time it takes for the orchestrator to post *MSG_ASR* (the message sent from the phone) to the blackboard.

MUF_Orchestrator_UtteranceFromASR, MUF_NLUDM_ReceivedMessage Time it takes for the NLU/DM component to receive *MSG_ASR* from the blackboard.

MUF_NLUDM_ReceivedMessage, MUF_NLUDM_ToBlackboard Time betwen the NLU/DM component receiving *MSG_ASR* and the client communication controller beginning to send *MSG_ASR* (to DialoguePython).

MUF_NLUDM_ToBlackboard, NLUDM_ASR Time it takes for *MSG_ASR* to travel from multiuser framework to NLU/DM.

NLUDM_ASR, NLUDM_NLU Time it takes for NLU/DM to convert ASR into a user intent; in other words, the duration of NLU.

NLUDM_NLU, NLUDM_DM Time it takes for NLU/DM to decide on action intent based on user intent; in other words, the duration of dialogue manager.

NLUDM_DM, NLUDM_Query Time it takes for movie recommendation to be queried from the movie database (query time).

NLUDM_Query, MUF_NLUDM_ReceivedAction Time it takes for action intent (with query result, if applicable) to travel from NLU/DM to multiuser framework.

MUF_NLUDM_ReceivedAction, MUF_SR_SetStrategy Time it takes for social reasoner to select strategy (stored as *MSG_SR*) based on *MSG_DM* (the action intent); in other words, the duration of SR.

MUF_SR_SetStrategy, MUF_NLG_ReceivedStrategy Time it takes for NLG component to receive *MSG_SR* from the blackboard.

MUF_NLG_ReceivedStrategy, MUF_NLG_Sentence Time it takes for BEAT to process the NLG sentence (generated using *MSG_SR*).

MUF_NLG_Sentence, MUF_Orchestrator_ToClient Time between the end of the BEAT process and its output (stored as *MSG_NLG*) being sent to the phone.

MUF_Orchestrator_ToClient, Phone_CCC_Request Time it takes for response (*MSG_NLG*) to travel from multiuser framework to phone.

Phone_CCC_Request, Phone_MultiuserEvent_Request Time between the client communication controller receiving the response and the multiuser event receiving the response.

Phone_MultiuserEvent_Request, Phone_MessageToUnity Time between the multiuser event receiving the response and the phone sending the message to Unity.

Phone_MessageToUnity, Phone_Unity_MessageFromAndroid Time between the phone sending the message to Unity and the start of the movie agent's response.

A.4 EntryTypePair Latencies

Pointer down to Google ASR output n: 42 min: 1960.0 max: 4940.0 mean: 2777.6190476190477 std dev: 752.6337804226018 median: 2425.0 skewness: 1.3381366163783324 kurtosis: 1.0584566345601547

Google endpointing n: 42 min: 160.0 max: 5900.0 mean: 1365.952380952381 std dev: 1350.8372462920183 median: 870.0 skewness: 1.7081486586609236 kurtosis: 2.5627220433963074

NLU/DM Query n: 42 min: 906.0 max: 2080.0 mean: 1342.357142857143 std dev: 291.7670990185261 median: 1267.0 skewness: 0.5813324047470905 kurtosis: -0.43922731187703157

Social strategy n: 42 min: 62.0 max: 333.0 mean: 216.4047619047619 std dev: 110.529779570454 median: 285.0 skewness: -0.47936878627777624 kurtosis: -1.7684598079312068

B Incremental Speech Results

B.1 Speech Service Latencies

n: 114 min: 0.0 max: 2170.0 mean: 487.6315789473685 std dev: 581.7828908547331 median: 80.0 skewness: 1.0273836716847586 kurtosis: 0.4274462709804938

B.2 Incremental Speech Result Times

Table 3 shows the results, where “timespan” refers to the elapsed time between the partial result and the full result.

B.3 More Incremental Things

Also, this will be a.

Table 3: Incremental Speech Result Times

Initial result time	Partial result time	Partial result text	Full result time	Full result text	Time between partial, full results
14:29:52.344	14:29:52.744	I like action	14:29:53.714	I like action movies	00:00:00.970
14:30:01.134	14:30:01.234	I don't have	14:30:01.424	I don't have any	00:00:00.190
14:30:11.394	14:30:11.884	n/a	14:30:11.884	I like Tom Cruise	00:00:00.000
14:30:23.714	14:30:24.964	I've already seen that	14:30:26.004	I've already seen that can you suggest something else	00:00:01.040
15:17:10.984	15:17:14.294	n/a	15:17:14.294	Yes absolutely I'm very much like horror movies	00:00:00.000
15:17:21.644	15:17:22.124	I don't really have	15:17:24.754	I don't really have any in terms of horror movies	00:00:02.630
15:17:33.314	15:17:33.314	Horror movie	15:17:33.344	Horror movies	00:00:00.030
15:17:46.634	15:17:48.784	n/a	15:17:48.784	Do you have something with Tom Cruise	00:00:00.000
15:18:00.914	15:18:03.134	Maybe a sci-fi movie	15:18:04.684	Maybe a sci-fi movie like with spaceships and stuff	00:00:01.550
15:18:19.264	15:18:19.264	Yes	15:18:20.514	Yes that's much better	00:00:01.250
15:18:27.824	15:18:29.264	n/a	15:18:29.264	Can you give me another one	00:00:00.000
15:18:41.444	15:18:42.254	I like romantic	15:18:43.134	I like romantic comedies	00:00:00.880
15:18:55.834	15:18:56.344	n/a	15:18:56.344	I like Woody Allen	00:00:00.000
15:19:02.854	15:19:04.564	n/a	15:19:04.564	How about Scarlett Johansson	00:00:00.000
15:19:30.794	15:19:30.794	No I don't	15:19:32.304	No I don't like Alec Baldwin	00:00:01.510
14:42:02.664	14:42:04.434	n/a	14:42:04.434	Give me some horror please	00:00:00.000
14:42:42.644	14:42:42.754	None	14:42:44.154	None I have none	00:00:01.400
14:42:49.214	14:42:49.374	I don't have	14:42:49.654	I don't have any	00:00:00.280
15:54:48.554	15:54:48.554	Action	15:54:49.004	Action movies	00:00:00.450
15:54:54.994	15:54:55.384	n/a	15:54:55.384	Quentin Tarantino	00:00:00.000
15:55:03.624	15:55:04.254	n/a	15:55:04.254	Charlize Theron	00:00:00.000
15:55:15.014	15:55:15.284	Already seen	15:55:15.524	Already seen that	00:00:00.240
15:55:26.704	15:55:26.704	I've seen	15:55:27.054	I've seen that too	00:00:00.350
15:56:04.894	15:56:05.044	Romantic	15:56:05.574	Romantic comedies	00:00:00.530
15:56:14.284	15:56:14.604	n/a	15:56:14.604	Nora Ephron	00:00:00.000
15:56:20.614	15:56:20.924	Nora Ephron	15:56:20.924	Nora Ephron Sarah	00:00:00.000
15:56:29.614	15:56:30.194	n/a	15:56:30.194	Judd Apatow	00:00:00.000
15:56:36.434	15:56:36.844	n/a	15:56:36.844	Meg Ryan	00:00:00.000
15:56:42.124	15:56:42.494	n/a	15:56:42.494	Tom Hanks	00:00:00.000
15:57:07.894	15:57:07.894	I've seen	15:57:08.164	I've seen that too	00:00:00.270
15:57:21.514	15:57:21.514	Thank	15:57:21.954	Thanks Sarah	00:00:00.440
16:04:39.414	16:04:39.414	Horror	16:04:39.544	Horror movies	00:00:00.130
16:04:48.714	16:04:49.124	n/a	16:04:49.124	I don't know	00:00:00.000
16:05:29.224	16:05:29.224	Thank	16:05:32.544	Thank you I like that movie	00:00:03.320
18:55:01.734	18:55:02.094	I don't have	18:55:03.514	I don't have any	00:00:01.420
18:55:26.044	18:55:26.484	Can you repeat	18:55:28.064	Can you repeat that	00:00:01.580
19:02:12.294	19:02:14.004	n/a	19:02:14.004	Christopher Nolan	00:00:00.000
19:07:04.714	19:07:04.714	I don't have	19:07:06.274	I don't have any favorite actors	00:00:01.560
19:08:39.134	19:08:39.344	I don't have	19:08:42.394	I don't have any favorite directors	00:00:03.050
10:29:10.875	10:29:11.305	Can you suggest	10:29:12.415	Can you suggest another one	00:00:01.110
10:29:33.575	10:29:33.575	Yes	10:29:36.615	Yes that's a good recommendation thank you Sarah	00:00:03.040
10:29:45.705	10:29:45.705	Thank	10:29:45.955	Thanks Sarah	00:00:00.250
10:40:41.305	10:40:41.725	I like action	10:40:41.975	I like action movies	00:00:00.250
10:40:52.095	10:40:52.215	I don't have	10:40:52.445	I don't have any	00:00:00.230
10:40:59.025	10:40:59.615	n/a	10:40:59.615	I like Tom Cruise	00:00:00.000
10:41:10.385	10:41:10.845	Can you suggest	10:41:11.935	Can you suggest another one	00:00:01.090
10:41:28.025	10:41:28.155	I've already	10:41:29.365	I've already seen that one	00:00:01.210
10:41:40.205	10:41:40.205	Thank	10:41:40.735	Thank you Sarah	00:00:00.530
16:15:53.985	16:15:55.735	n/a	16:15:55.735	Hi I like action movies	00:00:00.000
16:16:01.105	16:16:01.215	I don't have	16:16:01.425	I don't have any	00:00:00.210
16:16:07.685	16:16:08.165	n/a	16:16:08.165	I like Tom Cruise	00:00:00.000
16:16:17.935	16:16:17.935	I've already	16:16:19.095	I've already seen that one	00:00:01.160
16:22:10.095	16:22:11.815	n/a	16:22:11.815	Hi I like action movies	00:00:00.000
11:05:36.728	11:05:36.848	I don't have	11:05:37.098	I don't have any	00:00:00.250
11:05:43.618	11:05:43.778	I don't have	11:05:43.948	I don't have any	00:00:00.170