MVA: The Multimodal Virtual Assistant

Michael Johnston[1], John Chen[1], Patrick Ehlen[2], Hyuckchul Jung[1], Jay Lieske[2], Aarthi Reddy[1], Ethan Selfridge[1], Svetlana Stoyanchev[1], Brant Vasilieff[2], Jay Wilpon[1]
AT&T Labs Research[1], AT&T[2]
{johnston,jchen,ehlen,hjung,jlieske,aarthi,
ethan,sveta,vasilieff,jgw}@research.att.com

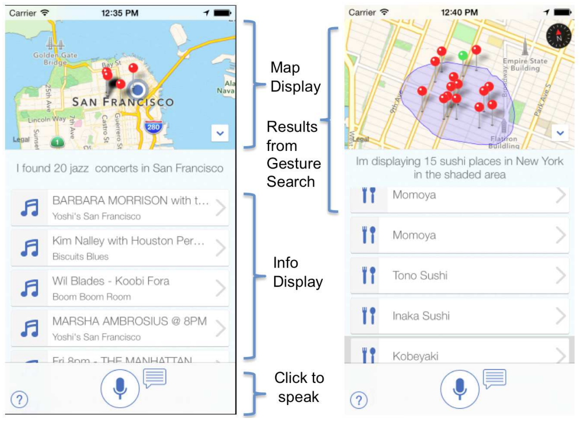
Abstract

The Multimodal Virtual Assistant (MVA) is an application that enables users to plan an outing through an interactive multimodal dialog with a mobile device. MVA demonstrates how a cloud-based multimodal language processing infrastructure can support mobile multimodal interaction. This demonstration will highlight incremental recognition, multimodal speech and gesture input, contextually-aware language understanding, and the targeted clarification of potentially incorrect segments within user input.

1 Introduction

With the recent launch of virtual assistant applications such as Siri, Google Now, S-Voice, and Vlingo, spoken access to information and services on mobile devices has become commonplace. The Multimodal Virtual Assistant (MVA) project explores the application of *multimodal* dialog technology in the virtual assistant landscape. MVA departs from the existing paradigm for dialog-based mobile virtual assistants that display the unfolding dialog as a chat display. Instead, the MVA prototype situates the interaction directly within a touch-based interface that combines a map with visual information displays. Users can interact using combinations of speech and gesture inputs, and the interpretation of user commands depends on both map and GUI display manipulation and the physical location of the device.

MVA is a mobile application that allows users to plan a day or evening out with friends using natural language and gesture input. Users can search and browse over multiple interconnected domains, including music events, movie showings, and places to eat. They can specify multiple parameters in natural language, such as "Jazz concerts around San Francisco next Saturday". As users find interesting events and places, they can be collected together into plans and shared with others. The central components of the graphical user interface are a dynamic map showing business and event locations, and an information display showing the current recognition, system prompts, search result listing, or plans (Figure 1).



I found 20 jazz concerts in San Francisco

BARBARA MORRISON with t...
Yoshi's San Francisco

Kim Nalley with Houston Per...
Biscuits Blues

Wil Blades - Koobi Fora
Boom Boom Room

MARSHA AMBROSIUS @ 8PM
Yoshi's San Francisco

Fri 8pm - THE MANHATTAN

Im displaying 15 sushi places in New York in the shaded area

Momoya

Momoya

Tono Sushi

Inaka Sushi

Kobeyaki

Figure 1: MVA User Interface

Spoken input begins when the user taps a microphone button on the display. As the user speaks, incremental speech recognition results appear. In addition to enabling voice input, the microphone button also activates the map as a drawing canvas, and enables the user to combine speech with drawing in coordinated multimodal commands. For example, a user might say, "Movies playing tonight in this area" while simultaneously outlining a relevant area on the map. Or a user may say, "Restaurants" while drawing a line down a specific street. MVA determines the intent and disambiguates concepts in the input in order to return relevant results. MVA then responds to user input multimodally, by updating the display and using speech synthesis to summarize results, provide feedback, or make requests for clarification and additional information.

2 Sample Interaction

In Figure 2 we present a sample of interaction from MVA that illustrates some of its capabilities. The user starts with a spoken natural language query where they specify some constraints: the type of music (*jazz*), location (*San Francisco*), and time (*tomorrow*). The system gets low confidence on the location, so it constructs a targeted clarification for clarifying only that constraint. The user repeats the location, and then the system searches for events meeting the user's constraints. The user then reviews the results, and follows on with a refinement: "What about blues?". Even though many parameters in this query are underspecified, the system applies contextually-aware natural language understanding and interprets this as "Blues concerts near San Francisco tomorrow". After selecting a concert, the user then searches for a restaurant nearby. The location of the concert remains salient. The user follows up with a multimodal query combining speech and gesture to search for similar restaurants in an adjoining area.

U:	"Jazz concerts near San Francisco tomorrow."
S:	"Where did you want to see jazz tomorrow?"
U:	"San Francisco."
S:	"I found 20 jazz concerts in San Francisco tomorrow." [*Zooms map to San Francisco and displays pins on map and list of results*]
U:	"What about blues?"
S:	"I found 20 blues concerts in San Francisco tomorrow."
U:	[*Clicks on a concert listing and adds it to the plan*]
U:	"Sushi restaurants near there."
S:	"I found 10 sushi restaurants."
U:	"What about here?" [*Circles adjoining area on map*]
S:	"I found 5 sushi restaurants in the area you indicated"

Figure 2: Sample Interaction

3 System Architecture

Figure 3 shows the underlying multimodal assistant architecture supporting the MVA app. The user interacts with a native iOS client. When the user taps the microphone icon, this initiates the flow of audio interleaved with gesture and context information streamed over a WebSocket connection to the platform.

This stream of interleaved data is handled at the server by a multimodal natural language processing pipeline. This fields incoming packets of

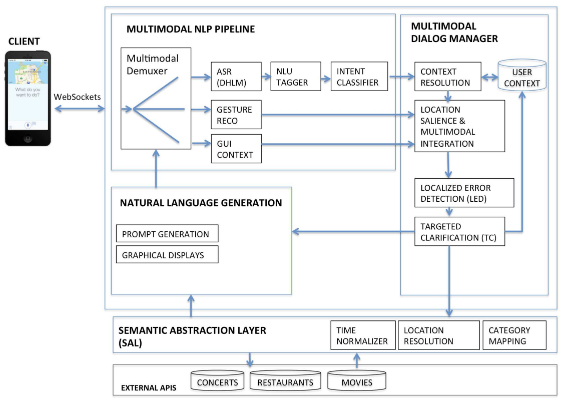


Figure 3: MVA Multimodal assistant Architecture

data from the client, demuxes the incoming data stream, and sends audio, ink traces, and context information to three modules that operate in parallel. The audio is processed using the AT&T Watson^{SM} speech recognition engine (Goffin et al., 2005). Recognition is performed using a dynamic hierarchical language model (Gilbert et al., 2011) that combines a statistical N-gram language model with weighted sub-grammars. Ink traces are classified into gestures using a linear classifier. Speech recognition results serve as input to two NLU modules. A discriminative stochastic sequence tagger assigns tags to phrases within the input, and then the overall string with tags is assigned by a statistical intent classifier to one of a number of intents handled by the system e.g. *search(music_event)*, *refine(location)*.

The NLU results are passed along with gesture recognition results and the GUI and device context to a multimodal dialog manager. The contextual resolution component determines if the input is a query refinement or correction. In either case, it retrieves the previous command from a user context store and combines the new content with the context through destructive unification (Ehlen and Johnston, 2012). A location salience component then applies to handle cases where a location is not specified verbally. This component uses a supervised classifier to select from among a series of candidate locations, including the gesture (if present), the current device location, or the current map location (Ehlen and Johnston, 2010).

The resolved semantic interpretation of the utterance is then passed to a Localized Error Detection (LED) module (Stoyanchev et al., 2012). The LED module contains two maximum entropy classifiers that independently predict whether a concept is present in the input, and whether a concept's current interpretation is correct. These classifiers use word scores, segment length, confusion networks and other recognition and context features. The LED module uses these classifiers to produce two probability distributions; one for presence and one for correctness. These distributions are then used by a Targeted Clarification component (TC) to either accept the input as is, reject all of the input, or ask a targeted clarification question (Stoyanchev et al., 2013). These decisions are currently made using thresholds tuned manually based on an initial corpus of user interaction with MVA. In the targeted clarification case, the input is passed to the natural language generation component for surface realization, and a prompt is passed back to the client for playback to the user. Critically, the TC component decides what to attempt to add to the common ground by explicit or implicit confirmation, and what to explicitly query from the user; e.g. "Where did you want to see jazz concerts?". The TC component also updates the context so that incoming responses from the user can be interpreted with respect to the context set up by the clarification.

Once a command is accepted by the multimodal dialog manager, it is passed to the *Semantic Abstraction Layer* (SAL) for execution. The SAL insulates natural language dialog capabilities from the specifics of any underlying external APIs that the system may use in order to respond to queries. A general purpose time normalization component projects relative time expressions like "tomorrow night" or "next week" onto a reference timeframe provided by the client context and estimates the intended time interval. A general purpose location resolution component maps from natural language expressions of locations such as city names and neighborhoods to specific geographic coordinates. These functions are handled by SAL—rather than relying on any time and location handling in the underlying information APIs—to provide consistency across application domains.

SAL also includes mechanisms for category mapping; the NLU component tags a portion of the utterance as a concept (e.g., a music genre or a cuisine) and SAL leverages this information to map a word sequence to generic domain-independent ontological representations/categories that are reusable across different backend APIs. Wrappers in SAL map from these categories, time, and location values to the specific query language syntax and values for each specific underlying API. In some cases, a single natural language query to MVA may require multiple API calls to complete, and this is captured in the wrapper. SAL also handles API format differences by mapping all API responses into a unified format. This unified format is then passed to our natural language generation component to be augmented with prompts, display text, and instructions to the client for updating the GUI. This combined specification of a multimodal presentation is passed to the interaction manager and routed back to the client to be presented to the user.

In addition to testing the capabilities of our multimodal assistant platform, MVA is designed as a testbed for running experiments with real users. Among other topics, we are planning experiments with MVA to evaluate methods of multimodal information presentation and natural language generation, error detection and error recovery.

Acknowledgements

Thanks to Mike Kai and to Deepak Talesra for their work on the MVA project.

References

Patrick Ehlen and Michael Johnston. 2010. Location grounding in multimodal local search. In *Proceedings of ICMI-MLMI*, pages 32–39.

Patrick Ehlen and Michael Johnston. 2012. Multimodal dialogue in mobile local search. In *Proceedings of ICMI*, pages 303–304.

Mazin Gilbert, Iker Arizmendi, Enrico Bocchieri, Diamantino Caseiro, Vincent Goffin, Andrej Ljolje, Mike Phillips, Chao Wang, and Jay G. Wilpon. 2011. Your mobile virtual assistant just got smarter! In *Proceedings of INTERSPEECH*, pages 1101–1104. ISCA.

Vincent Goffin, Cyril Allauzen, Enrico Bocchieri, Dilek Hakkani-Tur, Andrej Ljolje, S. Parthasarathy, Mazim Rahim, Giuseppe Riccardi, and Murat Saraclar. 2005. The AT&T WATSON speech recognizer. In *Proceedings of ICASSP*, pages 1033–1036, Philadelphia, PA, USA.

Svetlana Stoyanchev, Philipp Salletmayer, Jingbo Yang, and Julia Hirschberg. 2012. Localized detection of speech recognition errors. In *Proceedings of SLT*, pages 25–30.

Svetlana Stoyanchev, Alex Liu, and Julia Hirschberg. 2013. Modelling human clarification strategies. In *Proceedings of SIGDIAL 2013*, pages 137–141.