Understanding User's Cross-Domain Intentions in Spoken Dialog Systems

Ming Sun, Yun-Nung Chen and Alexander I. Rudnicky
School of Computer Science
Carnegie Mellon University
{mings,yvchen,air}@cs.cmu.edu

Abstract

Spoken language based intelligent assistants (IAs) have been developed for a number of domains but their functionality has mostly been confined to the scope of a given app. One reason is that it's is difficult for IAs to infer a user's intent without access to relevant context and unless explicitly implemented, context is not available across app boundaries. We describe context-aware multi-app dialog systems that can learn to 1) identify meaningful user intents; 2) produce natural language representation for the semantics of such intents; and 3) predict user intent as they engage in multi-app tasks. As part of our work we collected data from the smartphones of 14 users engaged in real-life multi-app tasks. We found that it is reasonable to group tasks into high-level intentions. Based on the dialog content, IA can generate useful phrases to describe the intention. We also found that, with readily available contexts, IAs can effectively predict user's intents during conversation, with accuracy at 58.9%.

1 Introduction

Intelligent assistants (IAs) on smart devices can converse with human users and help them with simple tasks such as finding a restaurant or sending a message. But human users may interact with their phones in more complex ways that can span multiple domains and apps. Planning and executing such multi-domain tasks is managed by users with the global context awareness for the tasks. We assume that IAs would become more valuable if they could actively assist on this more abstract task level, instead of treating each domain independently.

An app IA will, by definition, not maintain expectations of any follow-up domains. It is equally likely that the user would talk about food or weather next, regardless of the context. The consequences include 1) the system may not be fully prepared as it could be, if it understands how users actually structure such tasks; 2) the system may lose the opportunity to provide timely assistance to smoothly guide the dialog across domains ("Do you want to *send this picture to someone*?") or share the context of the multi-app interaction to the next elemental tasks ("You mean *the picture you just took*?"). In this paper, we address this issue by conducting a user study investigating how users perform multi-app tasks via language. We demonstrate that systems with shallow understandings, such as what the user said and the common task structures, can still improve interaction quality.

Systems that support multi-domain speech interaction have been studied in the past [11, 16]. One realization is a distributed architecture that allows different domains to be developed independently but cooperate with one another to respond to user input [8, 14, 2, 7]. However, these approaches wait for the user to invoke a domain by some explicit means and do not attempt to exploit relationships between applications that would exist in common tasks. We are interested in systems that become aware of these relationships and are able to proactively support user activities.

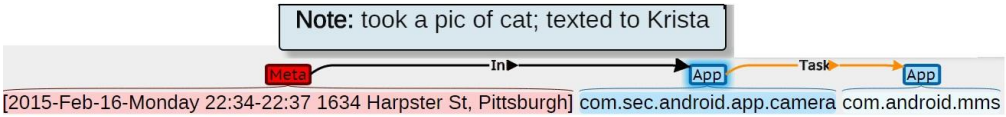

Figure 1: Multi-app annotation example; time and location are in red; constituent apps are blue.

It has been shown that based on simple context such as time or location, smartphones that anticipate a user's needs can significantly improve the efficiency of navigating the apps [19, 24]. However, language interaction may generate more information, allowing for better assistance. People also worked on discovering high-level latent structure in conversations [26], with the assumption that utterances (words) in a dialog are generated by a dynamic latent topic. In our work, we assume that we know previous apps (true for OS-level agent) and the latent topic (i.e. high-level intention such as "scheduling a meeting") is static within a dialog.

To investigate these questions we collected a corpus of actual smartphone usage, coupled with spoken language versions of these activities. We use these data to build models of user activity and provide an end-to-end pipeline for IAs to learn to understand and talk about such activities based on the language produced by the user. We evaluate how well IAs can anticipate potential task flow stemming from user intentions, by adapting to individual user's language and behavior patterns and surrounding context.

2 Data Collection

Participants in our study agreed to provide a continuous log of their smartphone use over an extended period of time. We built an Android app to log each app invocation as an event, together with date/time and the phone's location (if GPS is enabled). Participants were asked to upload their log on a daily basis. A privacy step allowed them to delete episodes that they might not wish to share.

The next task is to find meaningful groups of events (apps) from the logs submitted by users. However, due to the multi-tasking user behavior, time-based [20, 4] or content/lexicon-based [5] identification approaches may not come in handy [10]. However, this is not the focus of this paper. Initial analysis of the data indicated that phone usage could be segmented into episodes consisting of interaction events closely spaced in time. We used 3 mins inactivity to separate event clusters first, based on a pilot study. Participants were then invited to our lab on a regular basis (about once a week) to annotate their logs to provide more accurate groupings of events.

2.1 Smart Phone Annotation

We showed participants episodes extracted from their own logs via time-based approach. Meta information such as date, time, and location, was shown to aid recall. Participants were asked to produce two types of annotation, using the Brat server-based tool [21]: 1) **Task Structure**: link applications that served a common goal/intention; 2) **Task Description**: type in a brief description of the goal or intention of the task. For example, in Fig 1, the user first linked two apps (one about camera and another about text message) together since they were used for the goal of sharing a photo, and wrote a description "*took a pic of* ___". Some of the task descriptions were quite detailed and provided the actual app sequence executed (see example in Fig 1). However, others were quite abstract, such as "look up math problems" or "schedule a study session".

2.2 Interactive Dialog Task

We also let users talk to a Wizard-of-Oz dialog system to reproduce ("reenact") the multi-domain tasks in speech, instead of using the touch screen, in a controlled laboratory environment. The users were shown 1) apps used; 2) task description they provided earlier; 3) meta data such as time, location to help them recall the task (see the left part in Fig 2). The wizard (21-year-old male native English speaker) was instructed to respond directly to participant's goal-directed requests and to not accept out-of-domain inputs. The participants were not required to follow the order of the applications used on the smartphones. Other than for remaining on-task, we did not constrain expression. An example of a transcribed dialog is shown in the right part in Fig 2. This allowed us

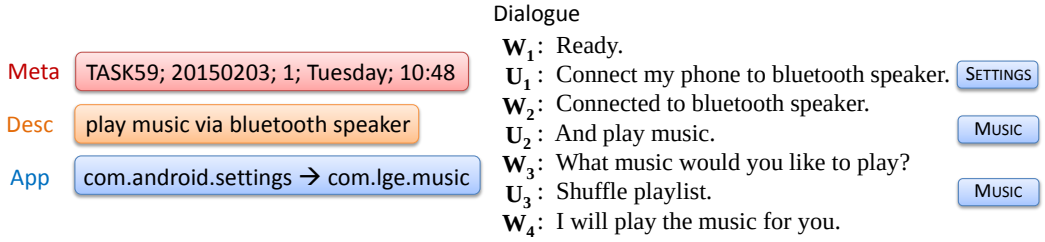

Figure 2: Multi-app task dialog example. Meta, Desc, App were shown to the participant. Utterances were transcribed manually or via Google ASR. Apps were manually assigned to utterances.

Category	#Participants	Age	#Apps	#Tasks	#Multi
Male	4	23.0	19.3	170	133
Female	10	34.6	19.1	363	322
Age < 25	6	21.2	19.7	418	345
Age ≥ 25	8	38.9	18.8	115	110
Native	12	31.8	19.3	269	218
Non-native	2	28.5	18.0	264	237
Overall	14	31.3	19.1	533	455

Table 1: Corpus characteristics. Age informally indicates young and old; #Apps is the average number of unique apps; #Multi is the number of multi-turn dialogues.

to create parallel corpora[1] of how people would use multiple apps to achieve an intention via both smartphone (touch screen) and language. We recruited 14 participants in this study and collected 533 parallel interactions, among which 455 involve multiple user turns (see Table 1 for the breakdown).

3 Context-Aware User Models

Context has been shown to improve the performance of interactive systems [17, 6]. To predict an individual user's next app (e.g., MUSIC) or current intention (e.g., "organize a dinner") during the conversation, we trained personalized user models that include the following contextual information: 1) **meta context**: time, day of week, location; 2) **behavioral context**: the previously launched app; 3) **language context**: words spoken by the user (e.g., "And play music" in Fig 2). These types of context are motivated by our observations; for example: a) people use ALARM more often in the morning on weekdays at home; b) for some user, CAMERA is more often followed by MESSENGER to share photos instead of EMAIL; c) "find the address of the Karaoke House in Oakland" not only indicates the use of BROWSER, but also hints that user may want to find the route to the address via MAPS.

However, using content-based *language* features such as words may suffer *vocabulary-mismatch* problem [10, 18], where statements related to the same topic may end up with non-overlapping terms, caused by minor differences such as misspellings, morphologies, etc. This can be addressed by enriching the queries (user input in our case) so similarity measurement can more easily capture the semantic relatedness [18]. In the current work we removed stop words and kept only lemmatized[2] verbs and nouns to reduce (morphology) noise. Semantic similarity is beyond the scope of the present work, although elsewhere we have used methods to expand a given (sparse) vocabulary [22].

The proposed IA can communicate at low-level actions such as "OK, let me first *find a restaurant in Oakland*" and at high-level intentions such as "I think you want to *plan a dinner*. Let me help you." We implemented this two-level communication by classifying input history to the predicted ranked list of apps or of intentions, respectively. The above three contextual ("meta") features are

[1] Dataset available at `http://www.cs.cmu.edu/~mings/data/MultiDomain.tar.gz`.
[2] `http://www.nltk.org/api/nltk.stem.html`

Feature	Train		Test	
	ACC	MAP	ACC	MAP
Language	60.5	66.5	39.1	44.0
Last App	41.9	50.7	29.7	37.2
Time	27.1	36.9	23.3	31.2
Day	26.7	36.2	22.8	30.7
Location	29.9	40.3	21.2	29.4
Majority	*25.8*	*35.2*	*23.9*	*31.7*
Meta	33.2	43.6	20.4	28.4
Meta+App	43.6	52.3	28.0	35.4
Lang+App	**59.2**	**65.3**	**40.0**	**45.0**
All	55.0	61.9	38.8	43.9

Table 2: App prediction

Feature	Train		Test	
	ACC	MAP	ACC	MAP
Last App	51.9	60.1	52.9	61.7
Language	44.6	53.6	39.3	50.5
Location	40.3	50.4	32.8	44.7
Time	31.5	42.4	31.5	44.4
Day	29.8	40.9	31.0	43.0
Majority	*27.4*	*38.1*	*31.7*	*44.4*
Meta	48.8	58.2	31.7	43.5
Meta+App	58.7	66.3	58.9	66.0
Lang+App	58.9	66.0	54.2	62.7
All	**64.5**	**71.1**	**58.9**	**66.1**

Table 3: Intention prediction

combined in a bag-of-words. For *time*, we use hours, from the 24-hour clock. For *day*, we used $\{weekday, weekend\}$, which appears to be more informative than $\{Monday, ..., Sunday\}$. For *location*, we use the street name instead of areas based on actual distances. (Note that users did not always share location). We evaluate using top-1 prediction accuracy (ACC) or mean average precision (MAP) over the ranked list reflecting two practical use cases [15].

4 Experiment and Results

In this section, we describe how the agent learns to predict: 1) the next app (low-level); 2) the current user intention (high-level). We also investigate features important for this functionality, and how to combine available features to further improve performance. For intention understanding, we introduce approaches to discover basic intentions and extract their semantics, to support the ability to interact on this intention level via natural language. We used each user's chronologically first 70% interactions as training data and used the remainder for testing.

4.1 Predicting Low-Level App

Results for different features across 14 participants' test data are shown in Table 2; we use multi-class logistic regression (individual features are ordered according to MAP on the test set). We used L_2 regularization and used 10-fold cross validation on the individual user's training set to determine the optimal regularization strength ($C \in [0.1, 10]$). The baseline, majority class, is also shown. Note that this is a difficult task; on average each participant has 19 unique apps. As we can see, Last App outperforms Meta features (time, location, day), which others have observed [19]; the Language feature (lemmatized verbs and nouns) is better than Last App. When we combine all individual features (All), we can improve on performance compared to just individual context.

Table 2 demonstrates that shallow understanding (e.g., what the user said and what app was launched earlier) can be predictive of a user's next actions. For example, given that MAPS is the predicted next app, the GUI can bring it into focus; the assistant can prompt "Would you like to *find the driving directions*?" or even proactively fetch and offer the information, e.g., "By the way, you can *take Bus XXX in five minutes*".

4.2 Modeling High-Level User Intention

We want IAs to be able to 1) discover meaningful intentions; 2) communicate with users at intention level and 3) predict the intention for ongoing conversation. We first cluster all past dialogs into a few groups, based on dialog content, task descriptions, and contextual information such as location, time. (Ideally) each group/cluster corresponds to a particular intent. Then IA can use the language produced in each cluster's dialogs to automatically propose key phrases that characterize the cluster semantics. Given a new interaction, IA would associate it with one of the clusters. In this work, we

Cluster	Item Examples (task descriptions supplied by participant)
1	"Picture messaging XXX", "Take picture and send to XXX"
2	"Look up math problems", "Doing physics homework", "Listening to and trying to buy a new song"
3	"Talking with XXX about the step challenge", "Looking at my step count and then talking to XXX about the step challenge"
4	"Playing [game] spiderman", "Allocating memory for spiderman"
5	"Using calculus software", "Purchasing Wolfram Alpha on the play store"
6	"Texting and calling XXX", "Ask XXX if she can talk then call her"
7	"Talking and sharing with group mates", "Emailing and texting group members"

Table 4: Intention clustering of tasks based on utterances, with typical descriptions.

describe this automated procedure, and also describe an interactive process to involve human users in refining clustering, and to provide natural language templates to carry phrases.

4.2.1 Intention Discovery

Similar to identifying tasks from search queries for Web Search Engines [10], our goal is to identify tasks/intentions from interaction data which is composed of sequence of apps, speech input and a task description. We cluster each participant's data into K clusters based on features including 1) apps being used; 2) words in description; 3) words in user utterances in the dialog.

The group of apps in a multi-app dialog is a natural hint of what the user is trying to achieve, except that the same set of apps may serve different purposes. For example, MAPS can given directions to certain places (navigation task) or provide review information/phone numbers for some business (restaurant reservation task). Using words in user descriptions ("finding a good restaurant") or the actual user utterances ("find me the nearest route to campus") may disambiguate the task identity.

For each user, the number of clusters K was determined by computing gap statistics, an unsupervised approach, during clustering [23] on the training set. On average, by using KMeans clustering, each participant has 7.6 ± 1.8 clusters. Examples (task descriptions) for clusters in Table 4 show that, in general, similar tasks cluster together. We asked 6 participants to evaluate the clustering performance on their own data. We showed them the members of each cluster—task description, full dialog, time and location. For each cluster, we asked them for agreement with the statement that "the dialogs shown are essentially the same task", from 1 (do not agree at all) to 5 (strongly agree). On average, these 6 participants rated their agreement with the system clustering with 4.2 ± 1.2 out of 5.0. Among 51 clusters altogether, these 6 users would further divide 10 of them (about 20%) into 27 subgroups. In short, people appear in general satisfied with the clustering algorithm we are using. It is possible that people would manually separate a certain amount of system-proposed clusters each into, on average, 2.7 subgroups. But this interactive process is optional.

4.2.2 Intention Representation in Natural Language

IAs may communicate with the user in language cast at the level of intention, especially in a clarification phase, e.g., "are you trying to *share picture with a friend*?" This involves template ("are you trying to ___?") and content ("*share picture* ___"). We can either elicit that information from the user or let the system automatically infer the semantics of the action in the cluster. Note that, by abstracting the semantics of the cluster that the input speech belongs to, IA can show true (mis-)understanding of the intention, compared with simply echoing the content of incoming speech.

Techniques are available for abstracting the semantics of each intention cluster. Text summarization may be used to generate high-level description [3, 9]. Keyphrase extraction is another alternative [25, 12, 1]. We chose the Rapid Automatic Keyword Extraction (RAKE) algorithm [1], an unsupervised, language-independent and domain-independent extraction method that has been reported to outperform other unsupervised methods such as TextRank [13] in both precision and F-score.

MANUAL	ASR	DESC	DESC+ASR	DESC+MANUAL
20.3	20.0	11.3	29.6	29.1

Table 5: Mean number of phrases generated using different resources

We adopted RAKE[3] toolkit to propose key phrases. We required that 1) each word has to have 3 or more characters; 2) each phrase to have at most 3 words; and that 3) each key word appear in the text at least once. These parameters were based on an examination of the data; we did not attempt tuning. We used 3 individual resources and 2 combinations. The three individual resources are task descriptions user typed in (DESC), manual transcription of user utterances in their dialogs (MANUAL) and their ASR transcriptions (ASR). The average number of key phrases that can be generated from each resource (or their combination) is shown in Table 5.

We asked the 6 users, reviewing their own clusters, to judge whether the system-generated phrase summarized all the activities in the cluster (binary judgement). For evaluation we used: 1) Precision at position K (P@K); 2) Mean Average Precision at position K (MAP@K); 3) Mean Reciprocal Rank (MRR). The first two metrics emphasize the quality of the top K phrases, MRR focuses on a practical goal—"how deep does the user has to go down a ranked list to find one useful phrase?". MRR is above 0.62 for the two combinations, that is, on average, the user will find an acceptable descriptive phrase in the top 2 items; an ANOVA did not show significant differences between resources. Using the more sensitive MAP@K and P@K metrics, DESC+ASR and DESC+MANUAL do best. The improvement becomes significant as K increases: a user-generated task description is very useful. Nevertheless, by observing user's speech commands or eliciting descriptions from user, IAs can generate understandable activity references, avoiding less efficient interactions (e.g., lists).

We also asked participants to suggest carrier phrases that could be used by the agent to refer to activities; these were unremarkable. Among the 23 phrases collected, "do you want to ___" and "would you like to ___" were the most popular. Likely content will remain the most important part.

4.2.3 Intention Prediction

Similar to app prediction described earlier, we built a user-dependent multi-class logistic regression model to predict user intention at each dialog turn. We used data from the 6 users mentioned above, whose clusters were initially proposed by the system automatically then refined by user if necessary. We evaluate the performance of each feature in Table 3. Again, L_2 regularization is applied with strength $C \in [0.1, 10]$ tuned through 10-fold cross validation. Last App outperforms other features. Best performance is obtained by combining all features. We observe improvement when language is incorporated (All vs. Meta+App), especially in the training set. Careful investigation is needed to understand the less improvement in the testing set.

5 Conclusion

We conducted a user study to investigate how human could interact with Intelligent Agents via speech in activities that span multiple domains/apps. We demonstrated that unsupervised techniques can be used for automatically identifying different types of high-level user intentions from dialogs, task descriptions, and other sources. We found that systems are able to develop a language that allows them to talk about intentions based on natural language extracted from what humans say. We evaluated methods to predict user intentions and the applications that they might subsequently invoke, by exploring user-specific app relationships, taking a hint from user utterances and using meta information such as time, location. We found language is predictive of app selection. Our findings provide preliminary evidence that IAs might be able to learn, over time, how to talk with users about activities that are organized at a level higher than that of individual applications; these would be on the level of activities and intents that are specific to the individual. We believe that this will be much like the functionality that currently is implemented in dialog applications, but customized to the needs of the individual.

[3] https://www.airpair.com/nlp/keyword-extraction-tutorial

References

[1] Michael W. Berry and Jacob Kogan. Text mining: applications and theory. 2010.

[2] Yun-Nung Chen, Ming Sun, and Alexander I. Rudnicky. Leveraging behavioral patterns of mobile applications for personalized spoken language understanding. In *Proceedings of 2015 International Conference on Multimodal Interaction (ICMI)*, 2015.

[3] Kavita Ganesan, Chengxiang Zhai, and Jiawei Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of COLING*, pages 340–348, 2010.

[4] Daqing He and Ayse Göker. Detecting session boundaries from web user logs. In *Proceedings of the BCS-IRSG annual colloquium on information retrieval research*, 2000.

[5] Daqing He, Ayse Göker, and David Harper. Combining evidence for automatic web session identification. In *Information Processing & Management*, 2002.

[6] Lee Hoi Leong, Shinsuke Kobayashi, Noboru Koshizuka, and Ken Sakamura. CASIS: a context-aware speech interface system. In *IUI*, pages 231–238, 2005.

[7] Qi Li, Gokhan Tur, Dilek Hakkani-Tur, Xiang Li, Tim Paek, Asela Gunawardana, and Chris Quirk. Distributed open-domain conversational understanding framework with domain independent extractors. In *Proceedings of SLT*, pages 566–571, 2014.

[8] Bor-shen Lin, Hsin-min Wang, and Lin-shan Lee. A distributed architecture for cooperative spoken dialogue agents with coherent dialogue state and history. In *Proceedings of ASRU*, 1999.

[9] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. Toward abstractive summarization using semantic representations. In *Proceedings of NAACL*, 2015.

[10] Claudio Lucchese, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Gabriele Tolomei. Identifying task-based sessions in search engine query logs. In *Proceedings of WSDM*, pages 277–286, 2011.

[11] Jean-Michel Lunati and Alexander I. Rudnicky. Spoken language interfaces: The OM system. *CHI91 Human Factors on Computing Systems*, 1991.

[12] Olena Medelyan. Human-competitive automatic topic indexing. In *Thesis Dissertation, University of Waikato*, 2009.

[13] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In *ACL*, 2004.

[14] Mikio Nakano, Shun Sato, Kazunori Komatani, Kyoko Matsuyama, Kotaro Funakoshi, and Hiroshi G Okuno. A two-stage domain selection framework for extensible multi-domain spoken dialogue systems. In *SIGdial Workshop on Discourse and Dialogue (SIGDIAL)*, pages 18–29, 2011.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] Alexander I Rudnicky, Jean-Michel Lunati, and Alexander M Franz. Spoken language recognition in an office management domain. In *Proceedings of ICASSP*, pages 829–832, 1991.

[17] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *Mobile Computing Systems and Applications*, pages 85–90. ACM, 1994.

[18] Xuehua Shen, Bin Tan, and ChengXiang Zhai. Implicit user modeling for personalized search. In *Proceedings of CIKM*, 2005.

[19] Choonsung Shin, Jin-Hyuk Hong, and Anind K. Dey. Understanding and prediction of mobile application usage for smart phones. In *ACM Conference on Ubiquitous Computing*, pages 173–182. ACM, 2012.

[20] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. Analysis of a very large web search engine query log. In *ACm SIGIR Forum*, 1999.

[21] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. BRAT: a web-based tool for NLP-assisted text annotation. In *Proceedings of EACL*, 2012.

[22] Ming Sun, Yun-Nung Chen, and Alexander I. Rudnicky. Learning OOV through semantic relatedness in spoken dialog systems. In *Proceedings of INTERSPEECH*, 2015.

[23] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. In *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2001.

[24] Akos Vetek, John A. Flanagan, Ashley Colley, and Tuomas Kernen. Smartactions: Context-aware mobile phone shortcuts. In *Human-Computer Interaction*, pages 796–799, 2009.

[25] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. Kea: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, 1999.

[26] Ke Zhai and Jason Williams. Discovering latent structure in task-oriented dialogues. In *Proceedings of ACL*, 2014.