Foodie Fooderson
A Conversational Agent for the Smart Kitchen

Prashanti Angara, Miguel Jiménez, Kirti Agarwal, Harshit Jain, Roshni Jain,
Ulrike Stege, Sudhakar Ganti, Hausi A. Müller
University of Victoria, Victoria, British Columbia, Canada
{pangara, miguel, kirti, harshit, roshni, ustege, sganti, hausi}@uvic.ca
Joanna W. Ng
IBM Canada Ltd., Markham, Ontario, Canada
jwng@ca.ibm.com

ABSTRACT

Conversational agents aim to offer an alternative to traditional methods for humans to engage with technology. This can mean to reduce the effort to complete a task using reasoning capabilities and by exploiting context, or allow voice interaction when traditional methods are not available or inconvenient. This paper introduces Foodie Fooderson, a conversational kitchen assistant built using IBM Watson technology. The aim of Foodie is to minimize food wastage by optimizing the use of groceries and assist families in improving their eating habits through recipe recommendations taking into account personal context, such as allergies and dietary goals, while helping reduce food waste and managing grocery budgets. This paper discusses Foodie's architecture, use and benefits. Foodie uses services from CAPRecipes—our context-aware personalized recipe recommender system, SmarterContext—our personal context management system, and selected publicly available nutrition databases. Foodie reasons using IBM Watson's conversational services to recognize users' intents and understand events related to the users and their context. We also discuss our experiences in building conversational agents with Watson, including desired features that may improve the development experience with Watson for creating rich conversations in this exciting era of cognitive computing.

CCS CONCEPTS

• **Human-centered computing** → *Ubiquitous and mobile computing systems and tools*;

KEYWORDS

Cognitive computing, conversational agent, smart kitchen, context management

ACM Reference format:
. 2017. Foodie Fooderson A Conversational Agent for the Smart Kitchen. In *Proceedings of 27th Annual International Conference on Computer Science and Software Engineering , Markham, Ontario, Canada, November 2017 (CASCON 2017),* 7 pages.

CASCON 2017, November 2017, Markham, Ontario, Canada

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

https://doi.org/10.475/123_4

1 INTRODUCTION

Foodie Fooderson (or Foodie for short) is a conversational kitchen assistant that uses IBM Watson technology to provide healthy recipes for cooks and reasons about dietary needs, constraints and cultural preferences. Foodie leverages services from CAPRecipes, our recipe recommender system [11], SmarterContext — our smarter context management engine [22], and public health and nutrition databases, including Spoonacular[1] and FoodEssentials.[2]

Conversational agents are applications that make use of natural language interfaces, such as text or voice, to interact with people, brands or services. Popular examples of such agents are Apple's Siri, Microsoft's Cortana, Google Assistant, Amazon's Alexa, and Mark Zuckerberg's Jarvis. These are typically built with the services provided by powerful commercial reasoning engines. They represent a new trend in digital gateways for accessing information, making decisions, and communicating with technology through sensors and actuators. The concept of conversing with a computing machine has been around for a long time. In 1966, Weizenbaum created ELIZA, the first natural language processing computing program [25]. ELIZA made use of directives along with pattern matching and substitution to respond to queries by humans. The idea of bots, such as ELIZA, was to not replace human intellect, but rather to have such tools as extensions of the human mind. We have come a long way since ELIZA. Natural language processing and artificial intelligence have advanced to such a degree that computers are able to almost accurately predict what a user's intentions are [10].

Many traditional command-line and graphical user interfaces are now giving way to conversational interfaces for a variety of applications. While the former rely on very specific input from the user, conversational interfaces make use of natural language understanding and infer the user's intent from linguistic sentences. Conversational agents are more than just conversational interfaces: they find practical application in areas where users need quick access to information, especially when the information is collated from different sources. In the new era of Cognitive Computing, it makes sense to have computers learn how to interact with us as opposed to us learning how to interact with computers [12].

[1] https://spoonacular.com/food-api
[2] http://developer.foodessentials.com/

This paper reports on our experience building smart kitchen assistants using cognitive technologies to augment human capabilities effectively.

A variety of commercial frameworks have been developed to provide services to define behaviours for conversational agents, including IBM's Watson Conversation,[3] Google's API.AI,[4] Amazon's Alexa,[5] Facebook's Wit.AI,[6] and Microsoft's Language Understanding Intelligent Service (LUIS).[7]

This paper discusses our experience with the IBM Watson Conversation platform in the area of home automation. Our conversational agent, Foodie, which manages a user's dietary needs and preferences, is augmented with contextual information provided by the user. Personal preferences, habits or health conditions (e.g., vegetarian, dislikes kohlrabi or soy allergy) are stored in personal context spheres (PCS) [22, 23]. Foodie considers users' personal information while making suggestions. The chosen application for our investigations, the kitchen, is an ideal environment for our purposes, since the domain is reasonably self-contained, but has complex contextual data—thus enabling us to research conversational cognitive agents. Ultimately, our vision for Foodie is to be a central hub of communication not just for the kitchen, but also for related activities such as grocery shopping or other similarly self-contained domains.

The rest of this paper is organized as follows. Section 2 gives an overview of Foodie's architecture. Section 3 describes how conversations are orchestrated between the user and Foodie, including goal articulation and recipe recommendations. Section 4 presents selected research related to context-aware, conversational agents. Section 5 discusses selected Watson services that are on our wish list. Finally, Section 6 concludes the paper and presents some avenues for future work.

2 FOODIE FOODERSON

Foodie is a cognitive conversational agent that augments the capabilities of home cooks by incorporating health-related information to aid one's eating habits. Foodie allows users to get recipe recommendations according to their contextual situation and taste preferences. For example, Foodie takes into account cultural background, time of the day, and cooking time. The latter being one of the most important factors among young adults interested in healthy eating [8]. It also distinguishes between the cases where the user has invited guests or cooks for her family.

Foodie provides recipe recommendation using CAPRecipes, our context-aware recipe recommendation engine. CAPRecipes receives the personal context and user preferences from Foodie, and recommends a list of recipes accordingly. The recipe recommendation engine is aware of the ingredients in the fridge and their details, including expiry date and quantity. These recipes are then filtered and sorted according to how much they affect the user's dietary goals and restrictions.

[3]https://www.ibm.com/watson/developercloud/conversation.html
[4]https://api.ai/
[5]https://developer.amazon.com/alexa-skills-kit
[6]https://wit.ai/
[7]https://www.luis.ai/

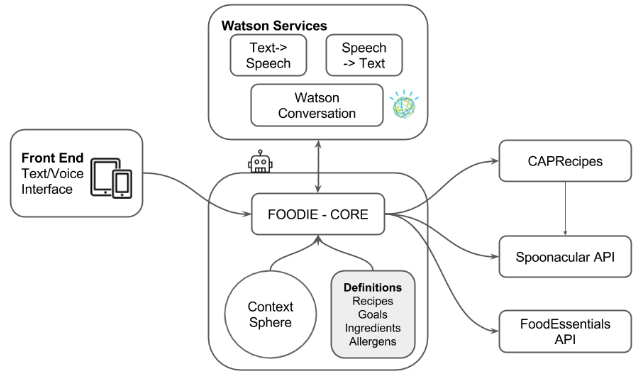

Figure 1: Foodie's architecture and dependencies

2.1 Functionality

The functionality of Foodie includes a variety of services that are useful for selecting ingredients and preparing a meal. Foodie's basic functionality is summarized below.

- *Update diet requirements and preferences:* Foodie provides a user interface that allows users to set and update parameters according to their diet and taste preferences. These parameters include: diet requirements (e.g., vegan, vegetarian), food allergies, excluded ingredients (e.g., dislikes), cuisine style, and budget.
- *Set a dietary goal:* Users may ask Foodie to add goals through a voice command. For instance, "Foodie, I would like to decrease my weight by 5%."
- *Set a medical condition:* Users may ask Foodie to set a new medical condition, such as diabetes and obesity, although this is out of scope for our current prototype.
- *Check available ingredients:* Users may ask Foodie for details about the current ingredients in the fridge (e.g., when a certain product expires or how much of a product is left). Ideally, Foodie would be connected to a Smart Fridge that provides services to obtain such details.
- *Ask for recipe recommendations:* Users may ask at any time for new recipe recommendations, considering the aforementioned constraints.

Our proof of concept allows users to interact with Foodie via text or voice regarding recipe recommendations, nutrition information or setting and updating goals. Foodie requests information from back-end databases, such as Spoonacular and FoodEssentials, and Watson services through their REST API. Foodie also provides a standard interface to integrate third-party messaging applications like Slack.[8]

2.2 Architecture

This section describes the architecture of Foodie as depicted in Figure 1. We describe the design of its conversational interface, dietary goals and recipe recommendations in Section 3.

Users ideally interact with Foodie via a hybrid interface (i.e., voice and text). Foodie can be integrated with services, such as

[8]http://slack.com

Slack, Facebook Messenger, Google Assistant or Siri. In any case, text and voice serve as the basis of input at the front-end. If speech is used, then the IBM Speech-to-Text service is used for converting voice to text. Each textual sentence serves as a request to Foodie's core engine. This engine takes into account, along with the text input, the user's personal preferences from their Personal Context Sphere [23].

Foodie's core is connected to Watson's Conversation Platform (cf. Section 3). This platform provides services that can understand natural language input. Watson Conversation provides *workspaces* that are containers for the artifacts of a conversation (i.e., the dialog structure, relevant entities and intents). Input from the user is redirected to the Conversation Workspace, which responds by returning the intent of the sentence as a JSON Object.

Information from the user's personal context sphere and the user's conversation with Foodie is used to make appropriate requests to back-end systems. Foodie is connected to our recipe recommender system CAPRecipes (cf. Section 3.2), which in turn uses the food and recipe database Spoonacular and the allergy database FoodEssentials. API requests are sent to one or more of these systems based on the user's input as parsed by Watson Conversation.

An example scenario: If user Alice asks "I'd like to cook something," Foodie recognizes this to be an indication to start cooking. Let's assume Alice asked this question at 5:00 pm. We also know from the Alice's personal context that she is allergic to peanuts. From the context of the conversation—an intent to start cooking and the time—and Alice's personal context, Foodie sends a request to the recipe recommender to retrieve a peanut-free recipe that can be prepared as dinner.

3 SMART CONVERSATIONS

Foodie is connected to IBM Watson, which provides services for designing the structure of conversations (as a *workspace*) that take place between Foodie and a user.

The work flow for such a conversation is illustrated below:

- *User: I'd like to get some nutrition information.* From this sentence, Watson identifies that the intent of the conversation: to get some information on nutrition, i.e., **`nutrition_info`**.
- *Foodie: Sure, which recipe are you looking for?* This response corresponds to a conversation node triggered by the intent **`nutrition_info`**.
- *User: Chocolate Chip Cookie Ice cream.* This sends out contextual information regarding the recipe via **`<?input_text?>`**, a context variable.
- Foodie's core engine takes this input and sends a request to the Spoonacular API to find nutrition details for this particular recipe.

3.1 Conversation Building Blocks

The building blocks for designing conversations are *intents, entities* and *dialog*. Information between Foodie and our conversational workspace is passed using *context variables*. This subsection also provides a description of the major intents, entities, dialog nodes and context variables that we considered in Foodie's conversation design.

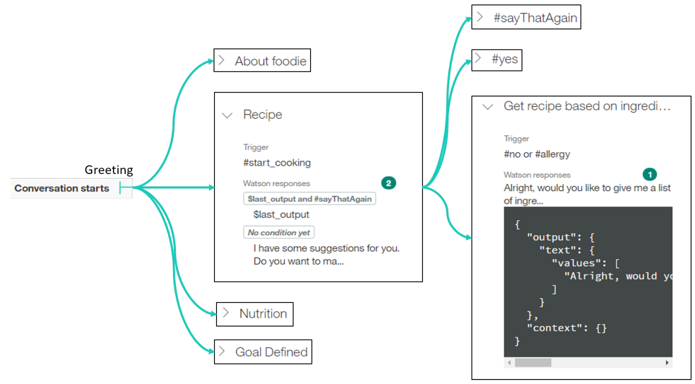

Figure 2: A subset of nodes in Foodie's Conversation

Intents: Intents define the purpose of the user's input. For example, if a user says "Can you suggest a recipe to me?" or "I'd like to cook something," Foodie recognizes here that the intention here is to retrieve a recipe. Our workspace on Watson Conversation classifies this intent as **`start_cooking`**. In our *workspace*, we define intents and different examples for each intent. This trains Watson to recognize intents of the user's input sentence with certain probabilities. Since Watson conversation comes with natural language understanding, when a user says "I'm hungry," even if this does not feature in the predefined examples for the intents, Watson correctly classifies this as a **`start_cooking`** intent. In a similar fashion, we define different intents for the different functionalities of Foodie. Some of them include the **`goal`** intent for setting, updating or removing the user's dietary goals or the **`nutrition_info`** intent that is defined for recognizing if the user has asked for nutrition information regarding a recipe or a product.

We define a few utility intents that help keep a smooth flow to the conversation. One such example is the **`say_again`** intent. Since Foodie supports voice, there may be instances where a user does not understand or hear what Foodie says. The **`say_again`** intent is recognized when a user says something along the lines of "Could you repeat that?" Another utility intent is **`reset`**, when the user wants to restart the conversation.

Entities: Entities are meant for keyword identification. Entities provide additional context to an intent. For example, there is a difference between "I want to eat" and "I want to eat a french dinner." Although the intent is recognized as **`start_cooking`**, there is additional information in the second sentence, **`french`** is a cuisine and **`dinner`** is a type of meal. Some of the entities in Foodie's conversation workspace are cuisine, type of meal, allergies, kind of diet, goals and goal types.

Dialog: The *dialog* is the third building block for a conversation. It provides the structure for possible flows in a conversation in the form of *nodes* connected by *directed edges*. These flows define how the application is to respond when it recognizes the defined intents and entities. Nodes are conditionally triggered usually based on intents or entities or a combination of both. Figure 2 shows a subset of the conversation nodes of Foodie. A conversation starts with a node for greeting (at Level 0). This is followed by nodes originating from the greeting node for the different things that a user may ask (these are the nodes at Level 1). For Foodie, we define nodes for recipe recommendations, nutrition information, allergy information and goals among others. Each of these nodes branches off to other nodes based on how the conversation flows. We also define a few utility nodes at each level that correspond to the utility intents. At every *level*, we define nodes for repeating (**say_again**), for resetting (**reset**) and for incomprehensible input (**anything_else**).

Context Variables: Context is a key part of the Dialog. Context variables provide the mechanism for passing information between the dialog and the application. For example, the following piece of JSON code represents contextual information that gets sent to Foodie.

Listing 1: Context Variables (JSON)

```
{
"context": {
 "request": "goal_defined",
 "goal": "@goal",
 "goalType": "@goalType",
 "expectation": "@expectation"
},
"output": {
 "text": {
  "values": ["@goalType the goal @goal
        by @expectation"]
  }
 }
}
```

The **context** key is used to pass the details of the request to Foodie. The **output** key is used to identify the output that is to be sent to the user (Foodie's response, if it exists, gets appended to this output). In the case of Foodie, decisions on the responses to the user are directed by the **request** key within the context.[9] The remainder of the keys serve as the parameters to this request. Listing 1 shows one such example of context variables within a dialog node. This node sends a request to Foodie requesting to add or change a user's goal details according to its parameters (i.e., @goal, @goalType and @expectation).

3.2 Recipe Recommendations

After the input is parsed by Foodie, relevant requests are sent to CAPRecipes. We choose CAPRecipes over other recipe recommender systems because of CAPRecipes' effective context management through user PCSs as well as the kitchen PCSs. Personal context includes aspects, such as health-related information (e.g., age, weight, height, diseases, and allergies), faith or belief restrictions, cuisine style, browsing history, likes on social media, and recipe ratings. The kitchen context features ingredients available in the user's refrigerator and cupboards (i.e., through digital information from receipts provided by grocery stores) including their quantities and expiry dates. Given this context (personal and kitchen), CAPRecipes makes recommendations using collaborative filtering [19]. For example, if CAPRecipes knows that a certain product in the kitchen, say eggs, is expiring soon, it will recommend making an appropriate dish using eggs before suggesting other dishes, thereby reducing the user's food wastage and optimizing the use of groceries.

[9]Note: Context here is not to be confused with the Personal Context Sphere. Context here refers to the context variables passed between Foodie and the Conversation workspace. The Personal Context Sphere refers to the individual user preferences.

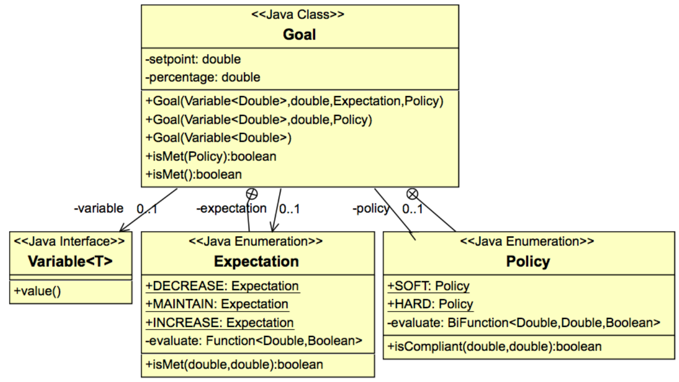

Figure 3: Class diagram representing dietary goals in Foodie

3.3 Dietary Goals

Figure 3 depicts the classes we use to model dietary goals. This abstraction allows us to model generic numeric goals associated with variables. Variables are continuous streams of measurements from the user's context. For example, the measurements from a weighing scale or an insulin pump are considered numeric variables that feed the necessary data to provide the user with accurate recommendations. When a user tells Foodie, "I would like to decrease my weight by 5%," it is translated into a goal whose *expectation* is to decrease, associated variable is *weight*, and policy is *soft*, meaning that the expected change may be 5% or more (this assumes that the user's context is already aware of their current weight, eating and exercise habits).

4 BACKGROUND AND RELATED WORK

There has been a steady increase in the number of applications that use conversations as a means of interacting with users. 2016 was dubbed as the year of the chatbot, and one year later, chatbots are still going strong—according to a market research study by MindBowser, in association with the Chatbots Journal.[10] Today these conversational applications are being used for a number of tasks across industries such as E-Commerce, Insurance, Banking, Healthcare, Telecom, Logistics, Retail, Leisure, Travel, Media among others. This section summarizes the research related to conversational applications in general and home automation applications in particular.

A recent survey conducted by Zamora et al. [26] explored how conversational agents can find a place in routine daily lives. They make a valid and compelling point, saying that currently, chatbots/conversational agent usage reduces as the novelty wears off. Their research collected qualitative insights from a diverse number of participants about perspectives on digital personal assistants.

[10]https://chatbotsjournal.com/global-chatbot-trends-report-2017-66d2e0ccf3bb

Their findings suggest that the users expect these assistants to be smart, high performing, seamless and personable. The users like the idea of assistants getting to know their personal quirks. Other studies, such as Milhorat et al. [18] highlight in their work, that the digital assistants are yet to become personal. For Foodie to be more personable, we make use of a personal context sphere [23] that maintains a list of user preferences.

In the survey by Zamora et al. [26], users were asked what method of input they preferred. Speaking to a conversational agent was found to be best when the user was multi-tasking or had hands or eyes occupied. Typing seemed to be best when the activity was complex. Users also found that they preferred to interact with bots for common administrative/menial needs and for emotional needs to provide motivation. We took this into consideration while building Foodie. The kitchen is a place where a user is multi-tasking and their hands are occupied. In such an environment, one would want to interact using voice, and would also want answers to be reliable.

Klopfenstein et al. [15] observed that many of the conversational platforms avoid voice processing and choose text as the most direct and unambiguous form of communication. They also talk about studies on interaction with voice such as those described in [17] and [3], and how unexpected turns of phrase and simple misunderstanding from the users can lead to misunderstanding of context and breakdown of the conversation. However, speech is becoming a more powerful and reliable way of interacting with devices [16]. There have been breakthroughs in this area such as the speech recognition engine "Deep Speech 2," developed by Baidu, which recognizes spoken words and interprets users queries with high accuracy [1].

The conversational part of an agent can be retrieval based or generative based [20]. Retrieval Based Models use a repository of predefined responses and heuristic to pick an appropriate response based on input and context. Therefore, they do not respond well to out-of-context questions. Generative models are still primitive and use deep learning and machine translation to parse sentences. They generate responses from scratch from extensive training data and therefore are good for unseen data. However, their responses can be highly varied and unexpected [10] (e.g., Microsoft's twitterbot Tay whose responses became racist [2]). In the quest to augment human cognition, retrieval based systems currently are more applicable and reliable than generative based systems [20]. In our case, Foodie is built using IBM's Watson Conversation. The conversational skeleton is built as a retrieval based system. To recognize what a particular user's intent is, Watson uses natural language understanding.

As mentioned above, the major tech giants also have their own conversational agents. Using these assistants, users can perform many tasks, such as, set alarms and reminders, search for nearby restaurants, send text messages, and get real-time updates on weather, traffic, and sports. For home automation purposes in particular, Google Home and Amazon's Echo are quite popular [7]. Amazon's Alexa is an Internet-based voice assistant for home automation. It can accomplish tasks like switching lights off/on, playing music, and maintain thermostat. Google Home is another voice-based personal assistant driven by Google Assistant. They have built-in applications (i.e., apps), that give customized results (e.g., weather or news). However, for information unavailable in these apps, the assistants redirect a user to a collection of web results. In our view, more applications should be developed that can interact directly with a phone's or home's digital assistant as opposed to returning a web search result. Eventually, we would like Foodie to be a part of this ecosystem as a "health buddy" that can take over the conversation in areas related to its expertise.

Wanner et al. [24] presented KRISTINA, a knowledge-based conversation agent, capable of communicating in multiple languages, with users from different social and cultural backgrounds. It also takes into account the emotional sensitivity of the user, while modeling response to the conversation. Even though Foodie currently responds only in English, personability is one of our main objectives. Although not a current feature, we would like to augment Foodie with support for other languages as well.

A variety of IoT applications discussed by Cabrera et al. [5] and Kim et al. [14] stress the importance of voice and text based control for their devices. When an application involves a number of decentralized interconnected devices, it makes sense to have an interface that seamlessly understands a user's request. For example, in a home automation scenario, it is easier to say "Siri, turn on the lights," than opening up an application and pressing a few buttons to turn on lights. With IoT starting to play an important role in kitchens [6], Foodie could be connected as a seamless interface between a user and all the devices. For instance, Foodie could serve as an gateway for the prototypes developed by Ficocelli et al. [9] or Blasco et al. [4] which assist elderly people in the kitchen activities, such as retrieving and storing items, or acquiring recipes for preparing meals.

5 DISCUSSION

So far, we have described the architecture of our kitchen assistant Foodie, including its conversational engine and its connected components. Currently, a user can converse with Foodie via text or voice. Foodie can be asked about recipe recommendations, nutrition information and modifying goals. Foodie replies to the user taking into consideration the user's request and context. This section discusses our experiences building the conversational part of Foodie, focusing on how conversations could be improved between Foodie and the user and how, in general, conversational agent building could be improved. While building Foodie, we realized that it is easy to make simple, retrieval-based conversational bots that respond to a specific sequence of inputs; for example, it is easy to build a simple question/answer agent. Sophisticated conversational agents are much more challenging to build. Take for instance, a conversation where the user tries to modify an input that was mentioned previously. In terms of Watson Conversation, this would involve having connections from every node back to every other node.

As stated in the related work section (cf. Section 4), conversational agents need to go beyond the initial intrigue and find a place in routine daily lives [26]. To ensure this, we need to make the interactions between the user and the conversational agent smoother. Conversational agents need be less like a fancy command-line interface and more "conversational." The necessity of having better conversations becomes even more apparent when voice is involved as opposed to text [3, 15]. For example, there are instances when the user does not understand what Foodie says or vice versa. Based on our experience building Foodie, we would like to suggest some improvements to Watson Conversation that might aid the process of building better conversational agents. We realize that some research might be required to make these suggestions a reality.

Global Nodes: We define global nodes to be the nodes that get repeated often in a conversation. In the case of Foodie, the nodes **`say_again`** and **`reset`** are repeated at every level of the conversation, at every branch. At any point in the conversation, a user may say "What did you say?" or "Let's start over," especially with the input/output is in voice. The number of these nodes increases exponentially as the conversation branches out and becomes complicated. Having global nodes by default would be nice to have in platforms like Watson Conversation.

Conditional Jumps: Jumps are a feature in Watson Conversation that allow going from a node in one branch to a node in another branch at any level. When a jump is configured at the end of a node, the conversation transitions to another node regardless (i.e., jumps are not yet conditional). At many points during the design phase of our conversation, we found that conditional jumps would have made the conversation easier to structure. For example, in the middle of recipe recommendation, a user may ask for allergy information for one of the ingredients.

Switching Context: The notion of conditional jumps is closely related to switching context. As discussed above, users may want to change something in the input or switch topics in the middle of a conversation. While it is possible to do this using jumps (better, conditional jumps), adding support to context switching at every node is cumbersome. Support for context switching would be a great addition to the platform.

Entity Negation: In Foodie's conversation workspace, we describe many entities like goals, allergens, meal types and cuisine. When a user says "I'd like to prepare a snack," the meal type "snack" is recognized. However, when a user says "I'm hungry, can you suggest something that is not a snack?," the meal type is still recognized as snack. The problem is similar to that of context switching or global nodes; it is something that gets repeated often and even though setting it up is possible, it is cumbersome to do so.

6 CONCLUSIONS AND FUTURE WORK

IBM advocates that we are in the era of cognitive systems: these are systems that learn at scale, reason with purpose, and interact with humans naturally. This makes the interaction between man and machine more co-operative [13]. In this paper, we discussed conversational agents and how they are changing the landscape of user engagement in this era of cognition. We did this by introducing Foodie, a smart conversational agent that assists users in the kitchen. Foodie is augmented with a personal context sphere that stores user preferences, which in turn leads to better engagement. We described how conversations between Foodie and the user are orchestrated using Watson conversation, the importance of having richer conversations and how that would make conversational agents a truly useful tool to augment human cognition. We also pointed out the differences in conversational structure between text and voice and stressed on how conversational platforms have to be improved to support the latter. Foodie serves as a use case for modern conversational agents.

Conversational assistants are becoming popular in various other domains, such as health care. According to Tibken et al. [21], some medical staff can spend nearly 10% of their time with patients answering questions about lunch, physician credentials, and visiting hours. Therefore, Philadelphia hospital, introduced a voice assistant in the patients rooms to address some simple comfort measures. All such use cases and the concepts of Foodie described in this paper can be used for designing better conversational agents. As discussed in the related work section, we would like to see more applications that can interact directly with a phone's digital assistant rather than being an independent application. Eventually, we would like Foodie being a module that can be integrated with any of the digital assistants such as Siri or Google Assistant. Furthermore, we would like to connect Foodie to smart appliances and smart services on a plug-and-play basis. Our vision for Foodie is to be part of bolder concept we call *Cognitive IOT Recipe Maven*. This Maven comprises smart kitchen components, including a Smart Fridge, and smart grocery components, including scanning QR codes of grocery items and generating coupons on the fly. We envision Foodie to be the central agent in the Cognitive IOT Recipe Maven.

ACKNOWLEDGMENTS

This work was funded in part by the National Sciences and Engineering Research Council (NSERC) of Canada, IBM Canada Ltd. and the University of Victoria. The authors would like to thank IBM's Centre for Advanced Studies, Canada for supporting this project.

REFERENCES

[1] Amodei, D., Anubhai, R., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Chen, J., Chrzanowski, M., Coates, A., Diamos, G., Elsen, E., Engel, J., Fan, L., Fougner, C., Han, T., Hannun, A. Y., Jun, B., LeGresley, P., Lin, L., Narang, S., Ng, A. Y., Ozair, S., Prenger, R., Raiman, J., Satheesh, S., Seetapun, D., Sengupta, S., Wang, Y., Wang, Z., Wang, C., Xiao, B., Yogatama, D., Zhan, J., and Zhu, Z. Deep speech 2: End-to-end speech recognition in English and Mandarin. *CoRR abs/1512.02595* (2015).

[2] Baeza-Yates, R. Data and algorithmic bias in the web. In *Proceedings 8th ACM Conference on Web Science (Websci 2017)* (2016), ACM, pp. 1–1.

[3] Bastianelli, E., Nardi, D., Aiello, L. C., Giacomelli, F., and Manes, N. Speaky for robots: The development of vocal interfaces for robotic applications. *Applied Intelligence 44*, 1 (2016), 43.

[4] Blasco, R., Marco, Á., Casas, R., Cirujano, D., and Picking, R. A smart kitchen for ambient assisted living. *Sensors 14*, 1 (2014), 1629–1653.

[5] Cabrera, J., Mena, M., Parra, A., and Pinos, E. Intelligent assistant to control home power network. In *IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC 2016)* (2016), IEEE, pp. 1–6.

[6] Chen, J.-H., Chi, P. P.-Y., Chu, H.-H., Chen, C. C.-H., and Huang, P. A smart kitchen for nutrition-aware cooking. *IEEE Pervasive Computing 9*, 4 (2010), 58–65.

[7] Ebling, M. R. Can cognitive assistants disappear? *IEEE Pervasive Computing 15*, 3 (July 2016), 4–6.

[8] Escoto, K. H., Laska, M. N., Larson, N., Neumark-Sztainer, D., and Hannan, P. J. Work hours and perceived time barriers to healthful eating among young adults. *American Journal of Health Behavior 36*, 6 (2012), 786–796.

[9] Ficocelli, M., and Nejat, G. The design of an interactive assistive kitchen system. *Assistive Technology 24*, 4 (2012), 246–258.

[10] Hirschberg, J., and Manning, C. D. Advances in natural language processing. *Science 349*, 6245 (2015), 261–266.

[11] Jain, H. CAPRecipes: Context-aware personalized recipes recommender. In *Poster: 26th IBM Center for Advanced Studies Conference: Annual International Conference on Computer Science and Software Engineering (CASCON 2016),* (2016).

[12] Kelly III, J., and Hamm, S. *Smart Machines: IBM's Watson and the Era of Cognitive Computing*. Columbia University Press, 2013.

[13] Kelly III, J., and Hamm, S. Computing, cognition and the future of knowing. Tech. rep., IBM Corp. White Paper, 2015.

[14] Kim, Y., Oh, H., and Kang, S. Proof of concept of home IoT connected vehicles. *Sensors 17*, 6 (2017), 1289.

[15] Klopfenstein, L. C., Delpriori, S., Malatini, S., and Bogliolo, A. The rise of bots: A survey of conversational interfaces, patterns, and paradigms. In *Proceedings 2017 Conference on Designing Interactive Systems (DIS 2017)* (New York, NY, USA, 2017), DIS 2017, ACM, pp. 555–565.

[16] Knight, W. Conversational interfaces: Powerful speech technology from China's leading internet company makes it much easier to use a smartphone. *MIT Technology Review* (2016).

[17] McTear, M., Callejas, Z., and Griol, D. *The conversational interface: Talking to smart devices*. Springer, 2016.

[18] Milhorat, P., Schlogl, S., Chollet, G., Boudy, J., Esposito, A., and Pelosi, G. Building the next generation of personal digital assistants. In *1st IEEE International Conference on Advanced Technologies for Signal and Image Processing (ATSIP 2014)* (2014), IEEE, pp. 458–463.

[19] Schafer, J., Frankowski, D., Herlocker, J., and Sen, S. Collaborative filtering recommender systems. *The Adaptive Web* (2007), 291–324.

[20] Song, Y., Yan, R., Li, X., Zhao, D., and Zhang, M. Two are better than one: An ensemble of retrieval-and generation-based dialog systems. *arXiv preprint arXiv:1610.07149* (2016).

[21] Tibken, S. Thanks to AI, you'll never be ignored at a hospital again. Tech. rep., CNET, 2017.

[22] Villegas, N. M. *Context Management and Self-Adaptivity for Situation-Aware Smart Software Systems*. PhD thesis, Department of Computer Science, University of Victoria, Canada, February 2013.

[23] Villegas, N. M., and Müller, H. A. Managing dynamic context to optimize smart interactions and services. In *The smart internet*. Springer, 2010, pp. 289–318.

[24] Wanner, L., André, E., Blat, J., Dasiopoulou, S., Farrùs, M., Fraga, T., Kamateri, E., Lingenfelser, F., Llorach, G., Martínez, O., et al. Kristina: A knowledge-based virtual conversation agent. In *International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2017)* (2017), Springer, pp. 284–295.

[25] Weizenbaum, J. ELIZA—A computer program for the study of natural language communication between man and machine. *Communications of the ACM 9*, 1 (1966), 36–45.

[26] Zamora, J. Rise of the chatbots: Finding a place for artificial intelligence in India and US. In *Proceedings 22nd International Conference on Intelligent User Interfaces (IUI 2017)* (New York, NY, USA, 2017), IUI 2017 Companion, ACM, pp. 109–112.