Deep Learning for Database Mapping and Asking Clarification Questions in Dialogue Systems

Mandy Korpusik, *Student Member, IEEE*, and James Glass, *Fellow, IEEE*

***Abstract*—A dialogue system will often ask followup clarification questions when interacting with a user if the agent is unsure how to respond. In this new study, we explore deep reinforcement learning (RL) for asking followup questions when a user records a meal description, and the system needs to narrow down the options for which foods the person has eaten. We build off of prior work in which we use novel convolutional neural network models to bypass the standard feature engineering used in dialogue systems to handle the text mismatch between natural language user queries and structured database entries, demonstrating that our model learns semantically meaningful embedding representations of natural language. In this new nutrition domain, the followup clarification questions consist of possible attributes for each food that was consumed; for example, if the user drinks a cup of milk, the system should ask about the percent milkfat. We investigate an RL agent to dynamically follow up with the user, which we compare to rule-based and entropy-based methods. On a held-out test set, assuming the followup questions are answered correctly, deep RL significantly boosts top five food recall from 54.9% without followup to 89.0%. We also demonstrate that a hybrid RL model achieves the best perceived naturalness ratings in a human evaluation.**

***Index Terms*—Reinforcement learning, convolutional neural network, semantic embedding, crowdsourcing, dialogue system.**

I. Introduction

TODAY'S AI-powered personal digital assistants, such as Siri, Cortana, and Alexa, convert a user's spoken natural language query into a semantic representation that is then used to extract relevant information from a structured database. The problem with these conventional approaches is that they often rely heavily on manual feature engineering and a set of heuristics for mapping from user queries to database entries. There is a fundamental mismatch between how people describe objects, and how the corresponding entities are represented in a structured database. For example, when a person describes the food they have eaten, they are likely to say they had a slice of "toast," rather than a piece of bread. However, the matching entries in a food database are various types of "bread, toasted," which is not an exact word match, and to complicate matters further, there are numerous other potential database matches, including French toast, toasted cereal, and toasted nuts or seeds. Historically, researchers have dealt with this text mismatch through tokenizing, stemming, and other heuristics.

To avoid this pipeline of text normalization and word matching database lookup, we instead take the approach of feeding raw input to neural networks, and allowing the models to learn to handle the text mismatch internally. We demonstrate that neural models learn *semantic* representations of natural language, in order to map from a user's query to a structured database. Instead of parsing a user query into a semantic fame and translating it into a SQL query, we instead let the neural model learn on its own how to transform natural language input and database entries into points in a shared vector space, where semantically similar entities lie close to each other.

As the first instantiation of this research, we apply our models to the nutrition domain. Diet tracking has gained popularity recently, as many Americans have begun trying to eat more healthy to counteract the rising obesity rate in the United States [1], [2]. According to the CDC, 39.8% of US adults in 2015-2016 were obese, leading to an estimated medical cost of \$147 billion in 2008.[1] However, existing mobile applications for food logging are often too tedious for people to use consistently, requiring entering one food item at a time, so they give up without reaching their diet goals.[2] Two possible solutions are bar code scanning of packaged products and computer vision to determine nutrient contents from a photo of a meal. We propose a complementary approach to simplify the food logging process even further with natural language: a user describes their meal verbally, and the system automatically determines which foods (and nutrients) they ate.

In this work, we explore the difficult task of mapping a natural language meal description to a set of matching foods found in the United States Department of Agriculture (USDA) food database. The huge search space[3] presents a challenge for search algorithms that aim to find the correct USDA food—it is almost impossible to pick the right food from a single user input. Thus, the system needs to ask followup clarification questions to narrow down the search space (see Fig. 1 for an example dialogue). However, the system should not ask too many questions of the user, since that renders the system unusable. The followup questions must also be intuitive for humans (e.g., the system should

Manuscript received December 14, 2018; revised April 4, 2019; accepted May 6, 2019. Date of publication May 23, 2019; date of current version June 6, 2019. This work was supported in part by the NIH under Grant #R21HL118347 [Roberts & Glass], in part by the USDA under agreement no. 58-1950-4-003 with Tufts University [Roberts], Quanta Computing, Inc. [Glass], and in part by the DoD through the National Defense Science Engineering Graduate Fellowship Program [Korpusik]. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Imed Zitouni. *(Corresponding author: Mandy Korpusik.)*

The authors are with the MIT Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: korpusik@csail.mit.edu; glass@mit.edu).

Digital Object Identifier 10.1109/TASLP.2019.2918618

[1] https://www.cdc.gov/obesity/data/adult.html

[2] A WSJ article cites research that shows MyFitnessPal users who come within 5% of their weight goal will log their food, on average, for six consecutive days,

2329-9290 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

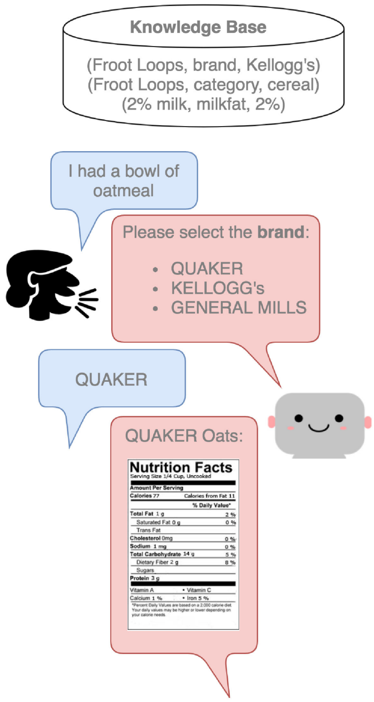


Fig. 1. In this sample dialogue, the AI agent asks a followup clarification question about the brand of oatmeal.

not ask about percent milkfat for vegetables), so the system needs to learn which attributes are reasonable to ask, for which foods. Hence, we must balance food recall with efficiency and ease of use—suitable for applying the reinforcement learning (RL) framework.[4]

We use supervised learning to train the initial food ranker [3]; however, when asking followup questions, it is unclear which order is best at each turn of the dialogue. We do not know ahead of time whether the questions asked will yield the optimal ranking until the dialogue is complete, and at the same time, we wish to finalize the ranking as quickly as possible to avoid annoying the user with excessive questioning. Thus, we investigate deep RL that relies on a reward function for determining the best order of food attributes to ask in order to narrow down the top-500 USDA hits as quickly as possible, while keeping the correct match ranked highest.

In our setup, we train a deep Q-network (DQN) to predict Q-values (i.e., the expected sum of future rewards) for actions (i.e., food attributes, such as the brand). In our experiments with a logistic Q-function, we verify that the DQN learns better value functions than linear approximators, as shown previously [4], [5]. The agent selects the question with the highest Q-value and asks the user to select the correct option for that attribute. The current ranked list of USDA hits gets updated accordingly, the new state is fed to the Q-network to determine the next best action to take, and so on, until the dialogue ends when there are no more questions left or fewer than five USDA hits remaining in the top-n ranked list.

We compare the RL agent to a rule-based model and an entropy-based solution. In addition, in some tasks, expert knowledge can help guide machine learning models (e.g., intuitively, it makes sense to ask how much milkfat is in milk, but not caffeine for vegetables). With this motivation in mind, we explore a new hybrid model for combining expert-engineered solutions with deep RL. We discover a tradeoff between asking fewer questions and achieving high food recall. Evaluated on humans, the rule-based model has fewest turns and lowest recall, while entropy has the most turns and highest food recall. Hybrid RL achieves a balance between the two, with 4.15 turns on average, 89.4% top-5 recall, and significantly better frustration and naturalness ratings.

The primary contributions of this work are:

1) Learning a shared semantic embedding space with a novel *end-to-end* neural network architecture, to map *directly* from natural language queries to structured database entries without manual feature engineering.
2) New, unpublished application of deep reinforcement learning to a novel, real-world dialogue task in the nutrition domain: narrowing down USDA food matches, given a user's meal log, by asking clarification questions.

II. Related Work

A. CNNs for Sentence Matching

While in our work we learn embeddings for USDA food entries through CNNs, recent work [6] has analyzed the relative strengths of various other sentence embeddings, including averaging word vectors learned with the continuous-bag-of-words method [7], LSTM auto-encoders [8], and skip-thought vectors based on gated recurrent units (GRU) [9]. Our approach differs from these in that we use a CNN rather than recurrent networks, and we learn the vectors through a domain-specific binary verification task for predicting whether a USDA food entry matches a meal description. While we do investigate RNNs, the reason we chose a CNN for our task is because it requires fewer parameters and is faster to train, while achieving similar performance,[5] and allow us to inspect the learned convolutional filters for interpretable patterns.

Similar work in learning joint embeddings for two different modalities or languages have explored a margin-based contrastive loss. For ranking annotations given an image, prior work directly incorporated the rank into the model's loss function, along with a hinge loss between true and false annotation samples [10]; similarly, a margin-based loss was used to learn a joint multimodal space between images and captions for caption generation [11], [12], and sentence/document embeddings

and other users for only three days: https://www.wsj.com/articles/new-reasons-why-you-should-keep-a-food-journal-1463419285.

[3]There are 5,124 food entries in the Standard Reference subset that we use and 215,557 branded food products.

[4]Our RL reward function favors short dialogues (ease of use) and followups that lead to the correct food (high recall).

[5]For both semantic tagging and database mapping, we have found that using the same architecture, but with an LSTM instead of a CNN, does not perform quite as well. For example, the Breakfast task for database mapping only achieves 61.8% top-5 recall with an LSTM, whereas our CNN model yields 64.8% top-5 recall.

were learned through a multilingual parallel corpus with a noise-contrastive hinge loss ensuring non-aligned sentences were a certain margin apart [13]. Other related work predicted the most relevant document given a query through the cosine similarity of jointly learned embeddings based on bag-of-words term frequencies [14].

Many researchers are now exploring CNNs for natural language processing (NLP). For example, in question answering, recent work has shown improvements using deep CNN models for text classification [15]–[17] following the success of deep CNNs for computer vision. Whereas these architectures take in a simple input text example and predict a classification label, our task takes in two input sentences and predicts whether they match. In work more similar to ours, parallel CNNs predict the similarity of two input sentences. While we process each input separately, others first compute a word similarity matrix between the two sentences (like an image matrix of pixels) and use the matrix as input to one CNN [18]–[20].

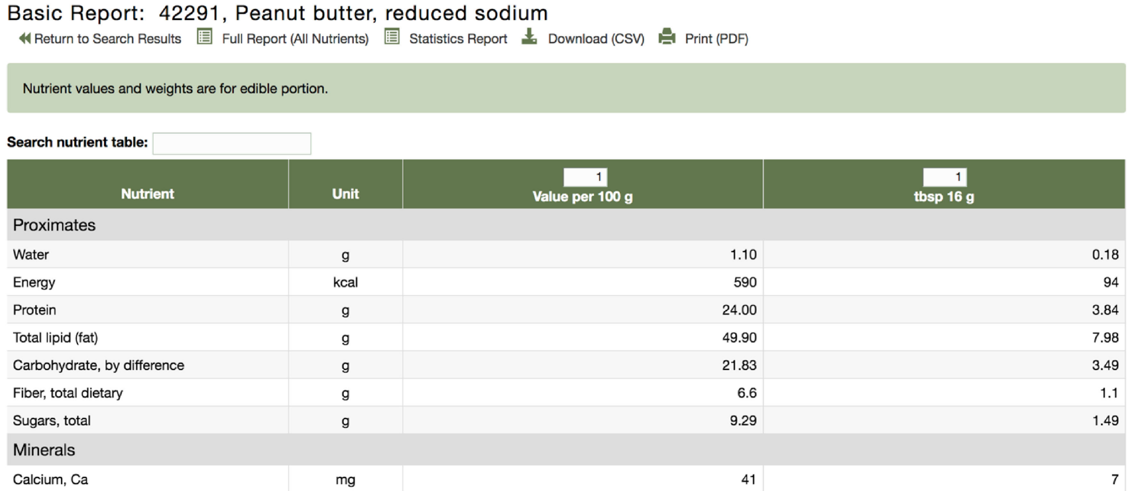
Attention-based CNN (ABCNN) has also been proposed for sentence matching. ABCNN [21] combines two approaches: applying attention weights to the input representations before convolution, as well as after convolution but before pooling. Our method is similar, but we compute dot products (our version of the attention scheme) with the max-pooled high-level representation of the USDA vector. Hierarchical ABCNN applies cosine similarity attention between CNN representations of a query and each sentence in a document for machine comprehension [22]. Thus, the attention comes after pooling across the input, whereas we compute the dot products between each meal token and the learned USDA vector. Finally, Korpusik *et al.* recently investigated CNNs for dialogue state tracking [23] and candidate response selection [24].

B. Deep Reinforcement Learning

Although RL has been used for many years in a wide array of fields, including dialogue systems and robot planning, only recently has deep RL begun to gain popularity among NLP researchers. Mnih *et al.*'s work on playing Atari games [25] led the shift from previous state-of-the-art Markov decision processes [26] to the current use of Q-networks to learn which actions to take to maximize reward and achieve a high score. While Mnih *et al.* used convolutional neural networks with video input for playing games, and Narasimhan *et al.* used deep RL for playing text-based games [5], the same strategy is also applicable to dialogue systems. Li *et al.* showed that deep RL models enabled chatbots to generate more diverse, informative, and coherent responses than standard encoder-decoder models [27]. Other work leveraged RL to construct a personalized dialogue system for a coffee-ordering task, where action policies were the sum of general and personal Q-functions [28]. Li *et al.* used RL in the movie domain to learn when to ask a teacher questions, and showed that the learner improved at answering factual questions about a movie when it clarified the question or asked for help or additional knowledge [29]. Similar to our hybrid model that balances a rule-based approach with RL, Henderson *et al.* used a hybrid supervised and RL technique trained on the COMMUNICATOR corpus for flight booking, although they found that a supervised approach outperformed the hybrid [30].

Deep reinforcement learning has also been successfully applied to task-oriented dialogue, which is similar to our diet tracking application. Zhao *et al.* used a deep Q-network to jointly track the dialogue state with a long short-term memory (LSTM) network and predicted the optimal policy with a multilayer perceptron to estimate the Q-values of possible actions [31]. In their case, the goal was to guess the correct famous person out of 100 people in Freebase by playing a 20 questions game and asking the user simulator questions relating to attributes such as birthplace, profession, and gender. While their questions are binary Yes/No questions, ours have many options to choose from. Another similar work responded to users with a matching movie entity from a knowledge base after asking for various attributes, such as actors or the release year [32]. Their system first learned from a rule-based agent and then switched to RL. They used a gated recurrent unit (GRU) network to track dialogue state and another GRU with a fully-connected layer and softmax on top to model policies. The main difference between our work and Dhingra *et al.*'s is they modeled uncertainty over database slots with a posterior distribution over knowledge base entities [32]. Williams *et al.* focused on initiating phone calls to contacts in an address book, and discovered that RL after pre-training with supervised learning accelerated the learning rate [33]. Peng *et al.* used a hierarchical deep RL model to conduct dialogues with multiple tasks interleaved from different domains [34].

Finally, Li *et al.* focused on optimizing all components in a task-oriented dialogue system simultaneously, in a single end-to-end neural network trained with RL for booking movie tickets [35]. In our case, the tagger and database ranker are separately trained components, and the RL policy is learned solely for asking followup clarification questions, without affecting the other steps in the pipeline. In future work, it would be interesting to explore jointly learning tagging, database mapping, and asking followup questions all in one model. In addition, since we do not have any dialogue data, we cannot train a user simulator and dialogue manager as is typically done. Instead of allowing open-ended responses from the user, the system provides a sample of possible options from which the user selects one. Therefore, the user simulator does not need to generate responses, and can be assumed to select the correct option each time (or we could introduce random noise if we wanted to simulate users occasionally answering questions incorrectly). Finally, the tagger in Li *et al.*'s work is an LSTM that jointly predicts intent and slots. In our work, the tagger is a CNN that only predicts slots, since there is only one intent (i.e., logging food). In summary, our RL agent for asking followup clarification questions is easily ported from one dialogue system and domain to another, where all of the components are already trained, rather than requiring the entire system to be trained from scratch, and our approach works even when no training dialogues are available.


Basic Report: 42291, Peanut butter, reduced sodium

Return to Search Results | Full Report (All Nutrients) | Statistics Report | Download (CSV) | Print (PDF)

Nutrient values and weights are for edible portion.

Search nutrient table:

Nutrient	Unit	1 Value per 100 g	1 tbsp 16 g
Proximates			
Water	g	1.10	0.18
Energy	kcal	590	94
Protein	g	24.00	3.84
Total lipid (fat)	g	49.90	7.98
Carbohydrate, by difference	g	21.83	3.49
Fiber, total dietary	g	6.6	1.1
Sugars, total	g	9.29	1.49
Minerals			
Calcium, Ca	mg	41	7

Fig. 2. A sample USDA food item with the food ID 42291 and database name "Peanut butter, reduced sodium."

III. Corpora

A. USDA Food Database

Our data is composed of two pieces—the structured food database, provided online by the United States Department of Agriculture (USDA),[6] and our natural language data. The USDA food database consists of 7,793 standard reference foods (e.g., produce, meat, bread, and cereal), as well as 239,533 branded food products (e.g., MCCORMICK & COMPANY, INC.). Each food item contains a unique ID, food name (consisting of a comma-separated list of descriptors, e.g., "Milk, 2%, without added vitamins"), and nutrition facts per serving size (see Fig. 2 for an example food item).

B. Natural Language Corpus

The second type of data we used to train our models was the natural language meal descriptions we collected via the crowdsourcing platform Amazon Mechanical Turk (AMT). Previously [36], we collected 22 k meal descriptions and their associated semantic annotations. However, the problem with this data is we did not know the correct USDA answers for each meal description. Here, we instead adopt a different strategy by asking Turkers to generate a meal description that matches a selected subset of USDA items. This enables us to build models that directly map from meal descriptions to USDA foods. We do not know the order or location of the foods in the meal description, but these can be inferred automatically by the neural network.

In order to generate reasonable meal description tasks, we partitioned the over 5 k foods in the USDA database into specific meals such as breakfast, dinner, etc. (see Table III). A given task was randomly assigned a subset of 97ndash;12 food items from different categories. To reduce biasing the language used by Turkers, we included images of the food items along with the less natural USDA titles (see Fig. 3). Turkers were asked to select at least three of the foods, and generate a meal description using these items (see Fig. 4 for the actual AMT interface). This enabled workers to select foods that would typically be eaten together, producing more natural meal descriptions and quantities.

IV. Database Mapping With Semantic Vectors

The natural language understanding component of a diet tracking system requires mapping a spoken or written meal description to the corresponding USDA food database matches. Following the work of [37], the first step consists of a convolutional neural network (CNN) followed by re-ranking to generate a ranked list of the top-n USDA foods ($n = 500$ in our experiments) per tagged food segment. We employed two steps to achieve a system that directly selects the best USDA matches for a given meal description: 1) we constructed a CNN model that learns vector representations for USDA food items through a binary verification task (i.e., whether or not a USDA item is mentioned in a meal description), and 2) we computed dot products over learned embeddings to rank the food database entries. The full system is depicted in Fig. 5.

A. Learning Semantic Embeddings With CNNs

As shown in Figure 6, our model is composed of a shared 64-dimension embedding layer, followed by one convolution layer above the embedded meal description, and max-pooling over the embedded USDA food name. The text is tokenized using spaCy.[7] The meal CNN computes a 1D convolution of 64 filters spanning a window of three tokens with a rectified linear unit (ReLU) activation. During training, both the USDA input's max-pooling and the CNN's convolution over the meal description are followed by dropout [38] of probability 0.1,[8] and batch normalization [39] to maintain a mean near zero and a standard

[6]https://ndb.nal.usda.gov/ndb/search/list

[7]https://spacy.io

[8]Performance was better with 0.1 dropout than 0.2 or no dropout.

Fig. 3. The instructions and example meal diaries shown to workers in our AMT data collection task.

deviation close to one. A dot product is performed between the max-pooled 64-dimension USDA vector and each 64-dimension CNN output of the meal description. Mean-pooling[9] across these dot products yields a single scalar value, followed by a sigmoid layer for final prediction.[10] This design is motivated by our goal to compare the similarity of specific words in a meal description to each USDA food.

To prepare the data for training, we padded each text input to 100 tokens,[11] and limited the vocabulary to the most frequent 3,000 words, setting the rest to UNK. We trained the model to predict each (USDA food, meal) input pair as a match or not (1 or 0) with a threshold of 0.5 on the output. The model was optimized with Adam [40] on binary cross-entropy loss, norm clipping at 0.1, a learning rate of 0.001, early stopping after the loss stops decreasing for the second time on the validation data (i.e., 20% of the data), and mini-batches of 16 samples. We removed capitalization and commas from the USDA foods.

[9]The inverse (mean-pooling before dot product) hurt performance.

[10]Note that our approach would work for newly added database entries, since we can feed the new database foods name into the pre-trained CNN to generate a learned embedding. This is the strength of using a binary prediction task, rather than a softmax output, so we do not have to re-train the network every time the database adds a new entry.

[11]We selected 100 as an upper bound since the longest meal description in the data contained 93 words.

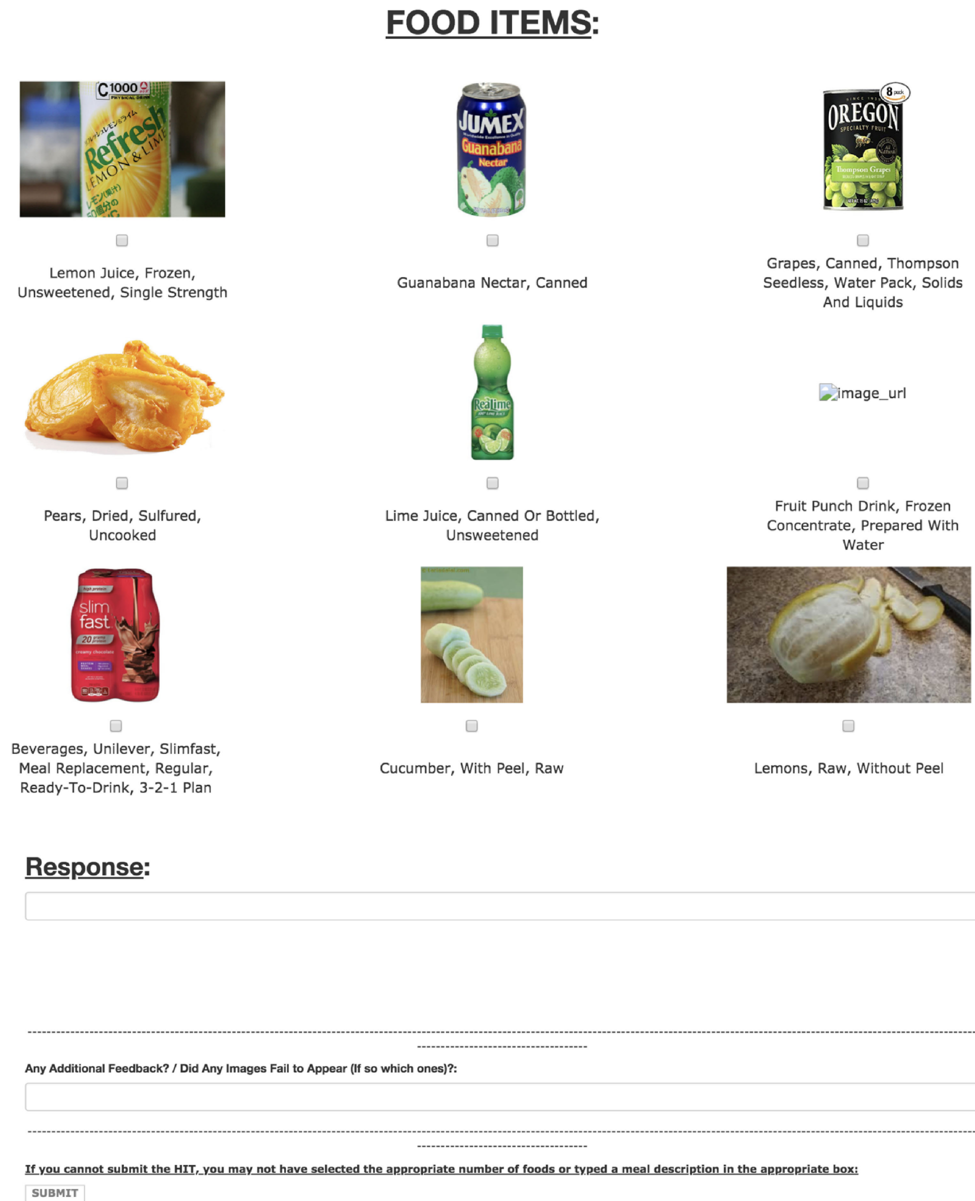


Fig. 4. The AMT interface, where Turkers had to check off at least three foods, and write a "response" (i.e., meal description containing those food items).

B. Semantic Tagging and Re-Ranking at Test Time

At test time, rather than feeding the entire meal description into the CNN, we instead first perform semantic tagging as a pre-processing step to identify individual food segments [36], [37], and subsequently rank database foods via dot products with their learned embeddings. A CNN tagger [41] labels tokens as Begin-Food, Inside-Food, Quantity, and Other. Then, we feed a food segment into the pre-trained embedding layer in the model described in Section IV-A to generate vectors for each token. Finally, we average the vectors for tokens in each tagged food segment (i.e., consecutive tokens labeled Begin-Food and Inside-Food), and compute the dot products between these food segments and each previously computed and stored USDA food vector.[12] The dot products are used to rank the USDA foods in two steps: a fast-pass ranking, followed by fine-grained re-ranking that weights important words more heavily. For example, simple ranking would yield generic milk as the top hit for 2% milk, whereas re-ranking focuses on the property 2% and correctly identifies 2% milk as the top match.

- *Ranking:* initial ranking of USDA foods using dot products between USDA vectors and food segment vectors.

[12] Our approach with CNN-learned embeddings significantly outperforms re-ranking with skipgram embeddings [42]. For comparison, on breakfast descriptions, our model achieves 64.8% top-5 recall, whereas re-ranking with skipgrams only yields 3.0% top-5 recall.

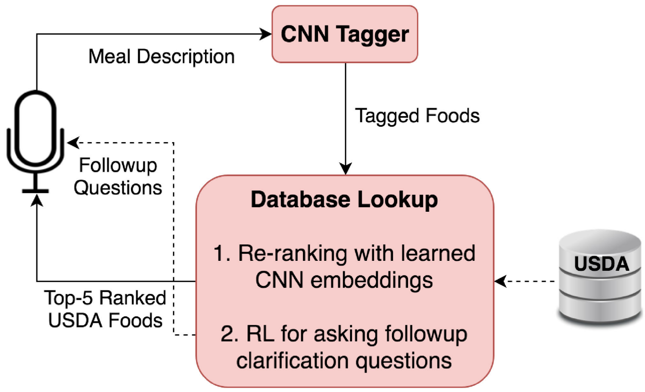


Fig. 5. The full system framework, where the user's meal is first tagged, then passed to the database lookup component, which consists of re-ranking based on learned CNN embeddings, followed by RL for asking followup clarification questions to narrow down the top-n ranked USDA hits to five.

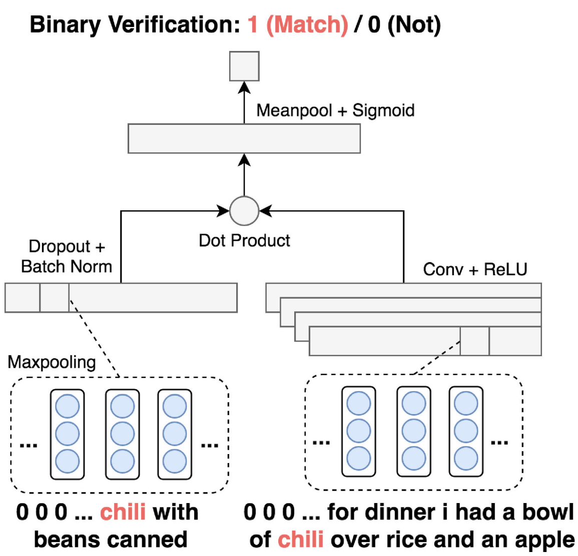


Fig. 6. Architecture of our CNN model for predicting whether a USDA food entry is mentioned in a meal, and simultaneously learns semantic embeddings.

- *Re-ranking:* fine-grained word-by-word *cos* similarity[13] ranking of the top-30 hits with weighted distance D [43].

$$D = \sum_i \alpha_i \max_j (w_i \cdot w_j) + \frac{1}{N} \sum_j \beta_j \max_i (w_i \cdot w_j) \quad (1)$$

where N refers to the length of the tagged food segment. The left-hand term finds the most similar meal description token w_j to each USDA token w_i, weighted by the probability α_i that token was used to describe the USDA food item in the training data. In the same way, the right-hand term finds the most similar USDA token w_i to each meal token w_j, weighted by the probability β_j that token w_j was used to describe that USDA food item in the training data (see Fig. 7).[14]

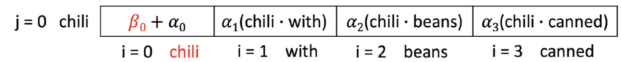


Fig. 7. A reranking example for the food "chili" and matching USDA item "chili with beans canned." There is only one β_0 term in the right-hand summation of equation 1, since there is only a single token "chili" from the meal description.

TABLE I
MEAL DESCRIPTION STATISTICS, ORGANIZED BY CATEGORY

Meal	# Foods	# Diaries	# Words per Diary
Breakfast	1167	4010	18.8
Dinner	2570	3850	21.6
Salad	232	4040	19.1
Sandwiches	375	4000	20.1
Smoothies	384	3850	20.1
Pasta/Rice	1270	4000	20.6
Snacks	1342	4077	19.1
Fast Foods/Meals	669	3886	19.1
All Foods	5124	31712	19.8

TABLE II
NEAREST NEIGHBOR FOODS TO THREE LEARNED USDA FOOD VECTORS

USDA Food	Nearest USDA Food
Rice white short-grain...	...Mexican Spanish Rice
Fast Foods Chicken Tenders	Chicken Broiler or Fryers...
Beans Baked Canned...	Beans Black Mature...

TABLE III
POSSIBLE ACTIONS (I.E., FOOD ATTRIBUTES) AVAILABLE TO THE MODEL, WITH EXAMPLE FOOD ITEMS FOR EACH

Attribute	Examples
Category	apples, cereal, pork, snacks
Name	apples, oatmeal, ham, popcorn
Brand	QUAKER, DOMINO'S
Type	instant, wheat, cheddar
Milkfat	nonfat, 1%, 2%, whole
Fat	low fat, reduced fat
Sweetness	sweetened, unsweetened
Addon	with beef, with added vitamin A
Salt	salted, unsalted, low sodium
Packing	regular pack, water pack
Preparation	raw, cooked, microwaved, boiled
Caffeine	with caffeine, no caffeine

C. Analysis of Learned Semantic Embeddings

Here we show through qualitative analysis that the CNN model is indeed learning meaningful vector representations of the USDA food entries, which is why it successfully maps from meal descriptions to USDA foods. If we look at the nearest neighbor to three USDA foods (see Table II) using Euclidean distance, we observe that the neighbors are in fact semantically similar. In addition, we can look at a t-SNE plot of the learned vectors for

[13] Note that for the initial ranking, we use dot product similarity scores, but for word-by-word similarity re-ranking, we use cosine distance. These distances were selected empirically, where on the dataset of all foods, these metrics yielded higher food recall than using dot products for both ranking steps, or cosine distance for both, and Euclidean distance was the worst.

[14] Although the sum appears biased toward longer USDA foods, the right-hand term is over each token in the food segment, which is fixed, and the left-hand term is normalized by the α weights. Dividing D by the number of tokens in the USDA food item hurt performance (39.8% recall on breakfast data versus 64.8% with the best model).

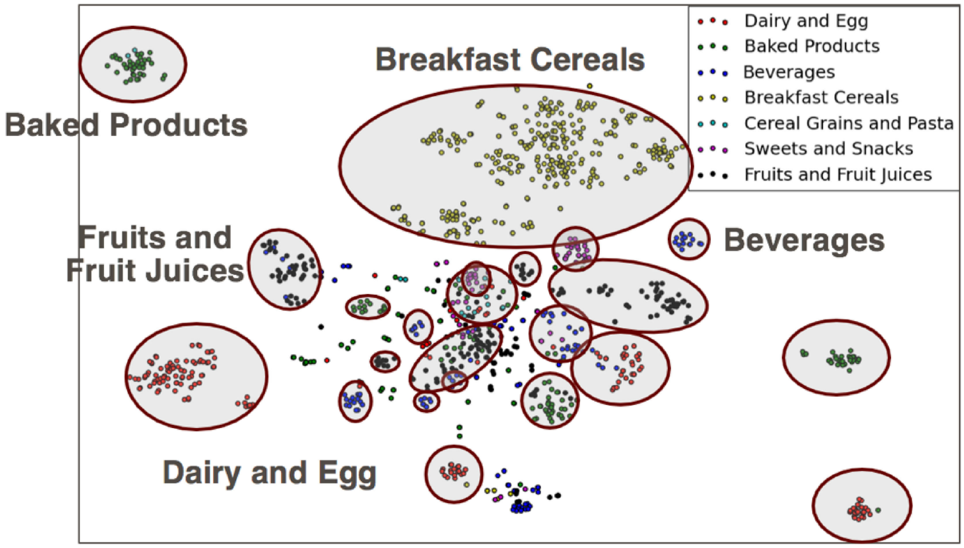


Fig. 8. A t-SNE plot of the learned embeddings for each USDA food database entry, where the color corresponds to the food category, and foods in the same category tend to cluster together. The center consists of several different clusters. The black points are fruits and fruit juices, and the dark blue are beverages. Since these are both drinks, they overlap. We hypothesize that the red cluster for dairy and egg that lies near the center is milk, since the center clusters are primarily related to beverages. In addition, just below the breakfast cereals, we note there are several teal points for cereal grains and pasta, which again has a reasonable location near cereals. The two small clusters of purple points, one which overlaps with breakfast cereals, and the other inside the cereal grains and pasta cluster, represent sweets and snacks. These may lie close to cereals since some sweets and snacks (e.g., Rice Krispie treats, Chex Mix) contain cereal.

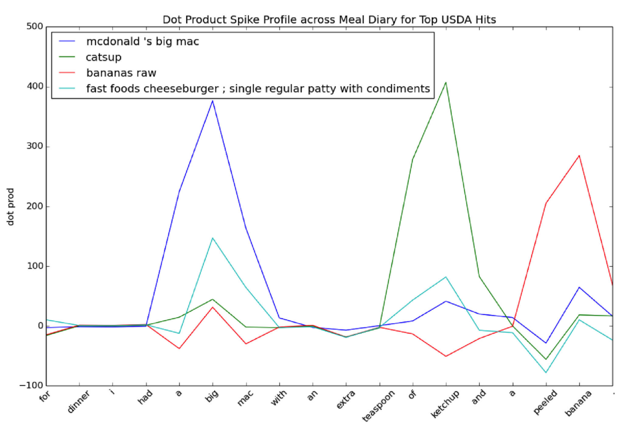


Fig. 9. Dot products between top USDA hits and meal tokens for the meal description *"for dinner I had a big mac with an extra teaspoon of ketchup and a peeled banana."*

each USDA food item, where a point is a single database food entry, and the color is the food category. In Fig. 8, we see that there are clusters corresponding to food categories (e.g., breakfast cereals are all yellow, baked products are green, and dairy and egg food items are red), which indicates that the learned embeddings for semantically related foods lie close together in vector space, as expected.

Looking at the model's predicted dot products between USDA foods and each token in a meal description, we observe spikes at tokens corresponding to that USDA food entry. We visualize the spike profile of the dot products at each token for the top USDA hits in Fig. 9. Despite the spelling mismatch, the USDA foods "McDonald's Big Mac" and "Fast Foods, Cheeseburger" spike at "big mac," "Catsup" peaks at "ketchup," and "Bananas, Raw" matches "peeled banana."

V. Asking Followup Questions With RL

The CNN re-ranking approach described in Section IV only yields 54.9% top-5 test set recall (i.e., how many times the correct USDA food appears in the top-5 ranking), but we would ideally achieve 90% top-5 recall in a system deployed to real users. Thus, in this work, we demonstrate that asking followup clarification questions about food attributes boosts top-5 recall to 89.0%. We investigated two hybrid reinforcement learning (RL) models: one using the rule-based method, and another based on entropy. The hybrid approach asks the first question according to the rule-based or entropy-based method's ordering, and afterward selects actions using the RL strategy. This method enables us to combine an expert-engineered solution (either starting with the most intuitive high-level question about category defined by hand-crafted rules, or computed using entropy), while also applying a deep Q-network to learn which attributes to ask next.

A. Rule-Based Followup

In this baseline approach, the dialogue agent asks about each food attribute in a pre-defined order: category, name, brand, type, milkfat, fat, sweetness, addons, salt, packing style, preparation method, and caffeine (see Table III). Any attributes for which the remaining hits all have the same value are skipped, since asking the value of these attributes would not narrow down the USDA hits any further.

B. Entropy-Based Followup

In a 20-questions-style task, such as that implemented by *Zhao & Eskenazi* to guess famous people in Freebase [31], an analytic solution based on an entropy measurement may be used to determine the optimal question to ask at each dialogue turn. In this scenario, Yes/No questions are asked (e.g., "Is it a man?") in an order chosen to minimize the number of questions required. Thus, the goal is to select the action that maximizes information gain (i.e., the reduction in entropy after asking a question), where entropy is defined as:

$$H(X) = \sum_{x \in X} P(x) \log_2 \frac{1}{P(x)} \tag{2}$$

However, in our work, instead of asking simple binary Yes/No questions, we are asking more complex questions that have multiple options as choices for the user to select from.[15] Instead of asking, "Was the brand Kellogg's?" we are asking, "Which brand was it?" We select the food attribute with the maximum entropy at each turn. For a given attribute, we define X as the set of possible values among the current top-n ranked foods (including null), and compute $H(X)$ via Eq. 2.

C. RL Followup

Since the rule-based approach always asks questions in the same order, we investigated whether a machine learning approach could figure out the best order of questions to ask in order to boost food recall further over the deterministic ordering. We do not know the optimal order until the end of the dialogue is reached and we can check whether the matching USDA hit was in the top-5 results, so we explored RL for this task because it uses delayed rewards computed at the end of each dialogue to update the model. As in a typical RL setup, the agent performs actions given the current **state** of the environment, and these **actions** result in **rewards**, which the agent learns from in order to choose future actions that maximize reward.

State The state s consists of the food segment's tokens (e.g., "bacon"); the USDA food IDs remaining in the narrowed-down, ranked list of matches; and the list of remaining actions, where each index is a possible food attribute, and the value at that index is 1 if the action has not been asked yet, or 0 if it has already been asked. For example, the binary action mask would be [0, 1, ... ,1] after one turn where the system asked about category (assuming the first action refers to the category attribute).

Action At each step, the RL agent must determine which action a to take by selecting one of the food attributes (see Table III) to ask a followup question about. Given state s_t, an action a_t is selected either randomly with probability ϵ, which decays from 1.0 to 0.01 at a rate of 0.995 in each minibatch, or as the argmax of the Q-network.

Reward r is defined as in [32]:

$$r = \begin{cases} -0.1 \times turn & \text{if dialogue not done} \\ 2(1-(rank-1)/5.0) & \text{else if rank} \leq 5 \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

where *turn* refers to the index of the followup question that is being asked, and *rank* is the final ranking of the correct food item (i.e., 1 is best).

The RL agent uses a two-layer feed-forward neural network (see Fig. 10) to estimate the Q-value (i.e., the expected sum of discounted future rewards). The dialogue is considered done when there are no more attributes remaining, or there are fewer than five USDA hits left to narrow down. Every 500 steps, the network gets updated based on a randomly sampled minibatch of 16 previously stored turns (i.e., using experience replay [31]).

[15] If we were to ask Yes/No questions, we would have to choose from 4,880 possible (attribute, value) pairs.

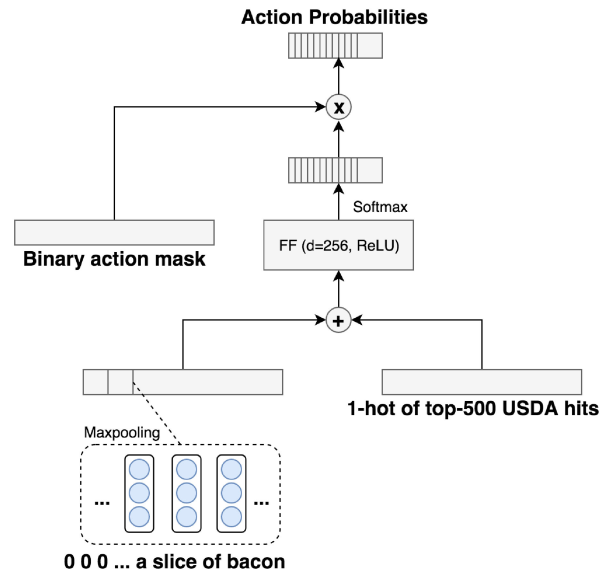


Fig. 10. The RL Q-network architecture, composed of a simple feed-forward (FF) layer followed by a softmax, which takes as input the tagged food segment (embedded and max-pooled), concatenated with a 1-hot vector of the top-500 ranked hits. The softmax output is multiplied by a binary action mask with zeros masking out all the previously selected actions and ones for the remaining actions. The first step is ranking all possible USDA food database matches, or hits, and selecting the top-n ($n = 500$). Each of the USDA foods is assigned an index in the 1-hot vector where the number of dimensions is the number of unique USDA foods, and the vector is initialized with zero. The foods that remain in the top-n ranking (starting with 500, and narrowed down after each turn) are assigned a value of one in the 1-hot vector.

The Q-network predicts Q-values for each action given the current input state s with a softmax layer. We define our policy as selecting the next action a either randomly with probability ϵ (i.e., exploration) or via the argmax of the predicted Q-values with probability $1 - \epsilon$ (i.e., exploitation). The loss for the Q-network, given chosen action a, is as follows, where discount factor $\gamma = 0.9$:

$$L = \frac{1}{2}(r + \gamma \max_{a'} Q(s', a') - Q(s, a))^2 \tag{4}$$

As described in Algorithm 1, during training, we first initialize the experience replay memory and Q-network. Then, for each training sample, we iterate through the cycle shown in Fig. 11 until the dialogue ends. Each dialogue begins with the user's food description (e.g., "a slice of bacon"), which is converted to the start state s_1. Given the current state s_t, an action a_t is selected either randomly with probability ϵ, which decays from 1.0 to 0.01 at a rate of 0.995 during each minibatch update, or as the argmax of the Q-network.

Using selected action a_t, the system follows up by asking about the chosen food attribute: "Please select the *category* for bacon: meat, oils..." and the user selects the correct attribute value (e.g., meat). The reward r_t is computed and new state s_{t+1} determined by narrowing down the remaining top-n USDA

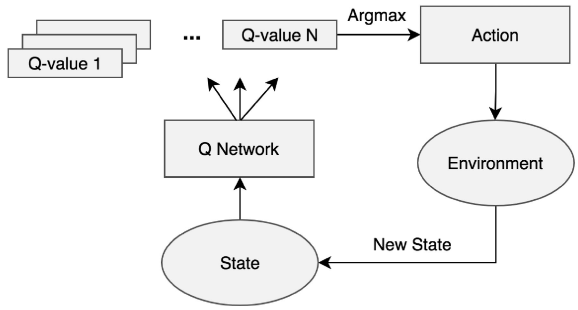


Fig. 11. The RL flow, where the current state is fed through the Q network to generate Q-values for each possible action. The action with the highest Q-value (or a random action) is selected, which is used to determine the reward and update the new state, continuing the cycle until the dialogue ends.

Algorithm 1: RL Training Algorithm.

Initialize experience replay memory $\mathcal{D}$
Initialize DQN parameters θ
for $i = 1, N$ **do**
  Initialize start state s_1 by ranking top-50 hits for meal
  while $\neg done$ **do**
    if $random() < \epsilon$ **then**
      Execute random action a_t
    else
      Execute action $a_t = \text{argmax}Q(s_t, a)$
    Observe next state s_{t+1} and reward r_t
    Determine whether dialogue is $done$
    Store memory $(s_t, a_t, r_t, s_{t+1}, done)$ in $\mathcal{D}$
    Sample random mini batch of memories $(s_j, a_j, r_j, s_{j+1}, done_j)$ from $\mathcal{D}$
    $y_j = \begin{cases} r_j & \text{if } done_j \\ r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta_t) & \text{else} \end{cases}$
    Perform gradient descent step on the loss $\mathcal{L}(\theta) = \frac{1}{2}(y_j - Q(s_j, a); \theta)^2$

hits based on the user's chosen attribute value. The experience $(s_t, a_t, s_{t+1}, r_t, done)$ is saved to the replay memory, where $done$ is a boolean variable indicating whether the dialogue has ended. The loop repeats, feeding the new state s_{t+1} into the Q-network again to generate Q-values for each action, until the dialogue ends and $done$ is true. The dialogue ends when there are five (or fewer) foods remaining, and the system returns the ranked list of USDA hits (e.g., "The results for bacon are: 10862–Pork, cured, bacon, pan-fried; 10998–Canadian bacon, cooked, pan-fried").

We use Adam [40] to optimize the Q-network, rectified linear unit (ReLU) activation and a hidden dimension size of 256 in the feed-forward layer, and 50-dimension embeddings. We process the user's raw input by tokenizing with the Spacy toolkit. Each token is converted to a vocabulary index, padded with zeroes to a standardized length (i.e., the longest food segment), and fed through an embedding layer mapping each token index to a 50-dimensional vector. Maxpooling selects the maximum value across all tokens for each dimension, resulting in a single 50-dimensional vector representation of the user's input. The 1-hot vector of top-500 hits is a binary list of all possible USDA hits in the food database, where each index corresponds to a food item, and the value at that index is 1 if that food is still in the ranked list, or 0 if not.

TABLE IV

FOOD RECALL WITH FOLLOWUPS, COMPARED TO A RE-RANKING CNN BASELINE WITHOUT FOLLOWUP [44]. STATISTICAL SIGNIFICANCE (T-TEST BASED ON FINAL RANK OF THE CORRECT FOOD) IS SHOWN, FOR HYBRID RULES-RL COMPARED TO OTHER METHODS (** FOR $p < .00001$)

Model	Turns	Top-1 Rec.	Top-5 Rec.
Re-ranking **	N/A	27.3	54.9
Entropy-based	6.00	**68.0**	**89.5**
Hybrid entropy	4.96	65.1	89.2
Rule-based **	**3.03**	59.4	89.1
Hybrid rules	4.15	66.4	89.4
Logistic RL **	3.60	64.6	88.4
Deep RL **	4.21	66.6	89.0

D. Experimental Results

For all our experiments, we used the written food descriptions and corresponding USDA food database matches collected on Amazon Mechanical Turk (AMT), as described in Section III, with 78,980 samples over the full USDA database of 5,156 foods, where Turkers wrote natural language descriptions about USDA foods [43]. To build the food knowledge base, we parsed each USDA entry and determined the values for the 12 manually defined food attributes (see Table III) with a set of heuristics involving string matching. We computed top-1 and top-5 USDA food recall (i.e., percentage of instances in which the correct USDA match was ranked first or in top-5) on 10% held-out test data.

E. Results From a User Simulator

We see in Table IV the performance of each of our expert-engineered and RL models for asking followups, along with the baseline re-ranking method taken from [44]. We also compare against a logistic regression baseline, which performs significantly worse than deep RL, illustrating why we need a deep Q-network with more than one layer. These results rely on a user simulator that is assumed to always choose the correct attribute value for the gold standard food item. The ground truth data consist of user food logs (e.g., "a slice of bacon") and matching USDA foods (e.g., 10862–Pork, cured, bacon, pan-fried). At each turn, the system asks the user to select an attribute value (e.g., "Please select preparation style for bacon: fried, baked..."), and the simulator selects the value for the correct USDA food (e.g., `preparation style=fried`).[16]

[16]In our work, if an attribute does not apply to a particular USDA food, it is assigned the `null` value, which is one solution for handling attributes that do not apply to all answer candidates.

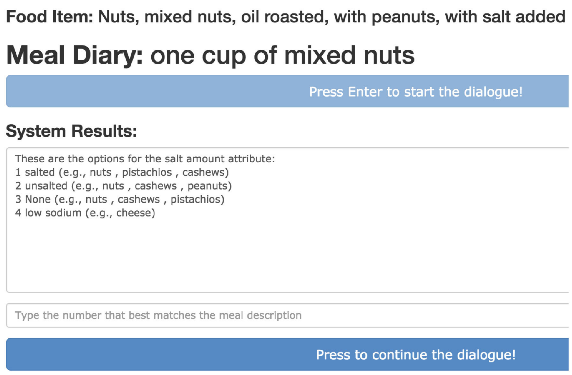


Fig. 12. An example interaction with the AMT task for online evaluation of the dialogue system.

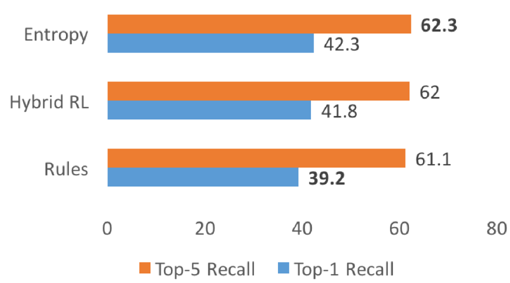


Fig. 13. Top-1 and top-5 food recall per model, evaluated on AMT.

All the followup approaches significantly outperform the baseline, boosting top-1 recall from 27.3% to 68.0% with entropy. The rule-based and entropy-based solutions are at opposite ends of the spectrum: using rules results in shorter dialogues, but lower recall, whereas the entropy solution has the highest recall, but longer dialogues. The RL agents find a balance between short dialogues and high recall. This tradeoff is similar to that in [45], where longer dialogues achieved higher precision with followup. This is because the rule-based method asks questions that have many possible attribute values as options, so when one of these options is chosen, the dialogue is already close to completion; however, since we limit the options shown to 10 to avoid overwhelming the user when the system is used by humans, the correct USDA food may be omitted, lowering food recall. The hybrid RL model strikes a balance between the rule-based method with fewer turns, and the entropy solution with high food recall.

F. Results From an Online Study With Humans

While the user simulator answers every question correctly, this is likely not true for humans interacting with the live system. Thus, we cannot assume the user will perfectly answer every question. In addition to maximizing recall, we want to minimize frustration while interacting with the system. Therefore, we conducted studies on Amazon Mechanical Turk to evaluate the various systems on both metrics, recall and ease of use, to confirm our hypothesis that the questions asked by the RL models are indeed the most intuitive and natural to people, enabling them to

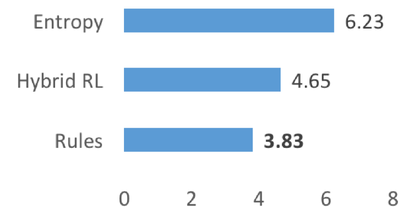


Fig. 14. Average turns per model, evaluated on AMT.

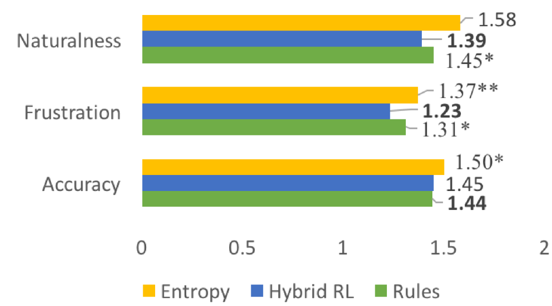


Fig. 15. Average perceived accuracy, frustration, and naturalness scores (lower is better) in the AMT study (* for $p < 0.05$, as compared to hybrid rules-RL).

TABLE V
EXAMPLE TARGET USDA FOODS FOR FIVE LEARNED FOLLOWUP PATTERNS ASKED BY THE RL MODEL

Attribute Followup	Example Foods
Brand	ON THE BORDER, taco ...
Brand	McDONALD'S Vanilla ...
Brand	Cereals, KASHI ...
Fat	Cookies ... reduced fat
Fat	Beef, ground, 93% lean meat ...
Fat	Cheese, ... low fat
Packing style ...	Fruit-flavored drink ...
Packing style ...	Plums, raw
Packing style ...	Tomato sauce, canned ...
Salt	Lentils ... without salt
Salt	Pickles ... reduced sodium
Salt	Cheese, parmesan, low sodium
... Addons	... with cheese and sausage
... Addons	... with added nutrients
... Addons	... with vitamin E added

answer questions more accurately and resulting in higher recall scores.

We incorporated our followup clarification dialogue system into a server and built an AMT task in which workers interacted with the system to narrow down the top-500 food options to the correct match, given the spoken food description and correct USDA food. We showed the possible attribute values at each turn, and displayed the top-10 food names per option (see Fig. 12). We paid $0.10 per task, and evaluated 1000 test samples for three models: rules, entropy, and hybrid rules-RL. We asked

Turkers to rate the interaction for perceived accuracy, frustration, and naturalness on a scale from 1 to 3, where 1 is the best and 3 is the worst.

In Fig. 13, we see the same pattern for food recall on humans as with the user simulator: the entropy-based solution has the highest recall scores, with hybrid RL in between, and rule-based the worst. We also see in Fig. 14 that the rule-based model has the shortest dialogues, again, and entropy-based longest. Interestingly, in Fig. 15, note that the hybrid rules-RL model has significantly better naturalness and frustration scores than rules and entropy, respectively. And despite entropy's high recall, Turkers rated it as the least accurate.

An analysis of the order of attributes asked by the RL model indicates a meaningful relationship with the target USDA food item described by the user. For example, in Table V, we see that the model asks about brands for branded food products; fat for meat, baked goods, and dairy; salt for vegetables and cheese; and addons for foods with added ingredients.

VI. Conclusion and Future Work

The contributions of this paper are two-fold: 1) we learned semantic vector representations of natural language with a novel model for mapping directly from natural language meal descriptions to matching USDA food database entries, and 2) we demonstrated the success of deep RL for asking followup clarification questions in the novel nutrition domain. Our neural approach to handling the text mismatch between messy natural language user queries and structured database entries both removes the need for manual feature engineering, and automatically learns *semantic* embeddings of natural language.

We compared RL to rule-based and entropy-based solutions for asking followup clarification questions, demonstrating a tradeoff between shorter dialogues and higher food recall. Asking followup questions significantly boosts recall, from 27.3% without any clarifications, to 66.4% top-1 recall with hybrid rules-RL followup. We also demonstrated that the hybrid RL model achieves the highest perceived naturalness and frustration scores when evaluated on humans.

In future work, we will add an option for users to speak or type a response to followup questions and will automatically narrow down the options as the user speaks or types. We also will make use of the attributes specified in the user's initial meal diary to avoid re-asking that attribute. Finally, we will investigate RL for online learning of users' preferences.

Appendix A
Sample Dialogues

We show sample interactions with the hybrid rules-RL model in Table VI for the meal "I had eggs with bacon and a glass of milk," where we assume the user has eaten fresh scrambled eggs, regular bacon (i.e., as opposed to meatless or low sodium), and a glass of 1% milk. We observe that, in general, the questions asked by the hybrid RL model seem fairly intuitive. For example, it asks about the preparation style of the eggs and bacon, as well as the milkfat and addons for milk (i.e., with or without added

TABLE VI
Sample Dialogue With the hybrid RL Approach, Which Selects the Correct Top-1 Hit for All Eaten Foods

I had eggs with bacon and a glass of milk.
System: Please select the **category** for: **eggs** • egg • sweets • meals
User: egg
System: Please select the **sweetness**: • None • sweetened
User: None
System: Please select the **preparation style**: • raw • dried • None • frozen • poached • fried • scrambled • hard-boiled • omelet
User: scrambled
Results for: **eggs** • 01132: Egg, whole, cooked, scrambled ✓
System: Please select the **category** for: **milk** • milk • snacks • sweets
User: milk
System: Please select the **preparation style**: • None • canned • dried
User: None
System: Please select the **addon**: • without added vitamins • with lauric acid oil • with added vitamins • None • with added calcium
User: with added vitamins
System: Please select the **milkfat**: • nonfat • None • whole • 2% • 1%
User: 1%
Results for: **milk** • 01082: Milk...1% milkfat, with added vitamins ✓ • 01104: Milk, chocolate...lowfat, with added vitamins
System: Please select the **category** for: **bacon** • meat • oils • vegetables
User: meat
System: Please select the **preparation style**: • None • cooked • fried • baked • heated
User: fried
Results for: **bacon** • 10862: Pork, cured, bacon, ...pan-fried ✓ • 43378: Pork, cured, bacon...pan-fried...reduced sodium • 10998: Canadian bacon, cooked, pan-fried

TABLE VII
SAMPLE DIALOGUE USING THE ENTROPY SOLUTION, WHICH ASKS MANY QUESTIONS, SOME OF WHICH ARE ODD, GIVEN THE FOOD (E.G., MILKFAT FOR EGGS). IF THE USER MISTAKENLY SELECTS SALTED INSTEAD OF `None`, ONLY RAW EGGS OR EGG NOODLES ARE RETURNED

I had eggs with bacon and a glass of milk.
System: Please select the **milkfat** for: **eggs** • None • whole
User: None
System: Please select the **fat**: • None • nonfat
User: None
System: Please select the **sweetness**: • None • sweetened
User: None
System: Please select the **salt**: • None • salted • unsalted
User: salted
Results for: **eggs** • 01160: Egg, yolk, raw, frozen, salted, pasteurized • 01255: Egg, whole, raw, frozen, salted, pasteurized • 20310: Noodles, egg, cooked, enriched, with added salt • 20510: Noodles, egg, cooked, unenriched, with added salt
System: Please select the **salt** for: **milk** • None • low sodium
User: None
System: Please select the **sweetness**: • None • sweetened • low calorie sweetener • unsweetened
User: None
System: Please select the **preparation style**: • None • canned • raw • frozen • dried
User: None
System: Please select the **fat**: • None • nonfat
User: None
System: Please select the **brand**: • None • MARS • HERSHEY'S • ROLO • CYTOSPORT • SYMPHONY
User: None
System: Please select the **milkfat**: • whole • None • nonfat • 1% • 2%
User: 1%
Results for: **milk** • 01175: Milk, fluid, 1% fat, without added vitamins • 01082: Milk...1% milkfat, with added vitamins ✓ • 01104: Milk, chocolate...lowfat, with added vitamins • 01088: Milk, buttermilk, fluid, cultured, lowfat
System: Please select the **fat** for: **bacon** • None • rendered fat
User: None
System: Please select the **sweetness**: • None • sweetened
User: None
System: Please select the **salt**: • None • low sodium
User: None
System: Please select the **preparation style**: • fried • baked • heated • cooked • frozen
User: fried
Results for: **bacon** • 10862: Pork, cured, bacon, ...pan-fried ✓ • 10998: Canadian bacon, cooked, pan-fried

vitamins). We show sample interactions with the entropy-based and rule-based models in Tables VII and VIII, respectively, for the same meal. Note the contrast between the short interactions with the rule-based model, versus the lengthy conversations with entropy.

TABLE VIII
SAMPLE DIALOGUE WITH THE RULE-BASED APPROACH, WHICH ASKS FEWER QUESTIONS, BUT YIELDS INCORRECT RESULTS FOR EGGS AND BACON DUE TO AMBIGUOUS NAMES (E.G., EGG VERSUS EGGS, AND BACON VERSUS PORK)

I had eggs with bacon and a glass of milk.
System: Please select the **category** for: **eggs** • egg • sweets • meals
User: egg
System: Please select the **name**: • egg • egg yolk • egg white • egg substitute • eggs
User: eggs
Results for: **eggs** • 43285: Eggs, scrambled, frozen mixture
System: Please select the **category** for: **milk** • milk • snacks • sweets
User: milk
System: Please select the **name**: • milk • ready-to-drink reduced fat beverage • chocolate milk • almond milk
User: milk
System: Please select the **type**: • whole • filled • dry • imitation • None • nonfat • human • fluid • reduced fat • lowfat • goat
User: lowfat
Results for: **milk** • 01082: Milk...1% milkfat, with added vitamins ✓
System: Please select the **category** for: **bacon** • meat • oils • vegetables
User: meat
System: Please select the **name**: • bacon • bacon bits • bacon and beef sticks • pork • turkey bacon • canadian bacon
User: bacon
Results for: **bacon** • 16104: Bacon, meatless

ACKNOWLEDGMENT

The authors would like to thank Karthik Narasimhan, Yonatan Belinkov, Mitra Mohtarami, and Tuka Al-Hanai for their advice.

REFERENCES

[1] Y. Wang and M. Beydoun, "The obesity epidemic in the United States–gender, age, socioeconomic, racial/ethnic, and geographic characteristics: A systematic review and meta-regression analysis," *Epidemiologic Rev.*, vol. 29, no. 1, pp. 6–28, 2007.

[2] World Health Organization. *Obesity: preventing and managing the global epidemic*. Tech. Rep. 894, World Health Organization, 2000.

[3] M. Korpusik, Z. Collins, and J. Glass, "Semantic mapping of natural language input to database entries via convolutional neural networks," in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, 2017, pp. 5685–5689.

[4] J. He *et al.*, “Deep reinforcement learning with an action space defined by natural language,” in *Proc. Workshop Track Int. Conf. Learn. Representations*, 2016. [Online]. Available: https://openreview.net/pdf?id=WL9AjgWvPf5zMX2Kfoj5

[5] K. Narasimhan, T. Kulkarni, and R. Barzilay, “Language understanding for text-based games using deep reinforcement learning,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1–11.

[6] Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg, “Fine-grained analysis of sentence embeddings using auxiliary prediction tasks,” 2016, arXiv:1608.04207.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, arXiv:1301.3781.

[8] J. Li, M. Luong, and D. Jurafsky, “A hierarchical neural autoencoder for paragraphs and documents,” 2015, arXiv:1506.01057.

[9] R. Kiros *et al.*, “Skip-thought vectors,” in *Proc. Advances Neural Inf. Process. Syst.*, 2015, pp. 3294–3302.

[10] J. Weston, S. Bengio, and N. Usunier, “Wsabie: Scaling up to large vocabulary image annotation,” in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, vol. 11, pp. 2764–2770.

[11] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3128–3137.

[12] D. Harwath, A. Torralba, and J. Glass, “Unsupervised learning of spoken language with visual context,” in *Proc. 30th Conf. Neural Inf. Process. Syst.*, 2016, pp. 1858–1866.

[13] K. Hermann and P. Blunsom, “Multilingual models for compositional distributed semantics,” in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 58–68.

[14] P. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, “Learning deep structured semantic models for web search using clickthrough data,” in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 2333–2338.

[15] A. Conneau, H. Schwenk, Y. Lecun, and L. Barrault, “Very deep convolutional networks for text classification,” in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics: Vol. 1, Long Papers*, 2017, pp. 1107–1116.

[16] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.

[17] Y. Xiao and K. Cho, “Efficient character-level document classification by combining convolution and recurrent layers,” in *Proc. 30th Int. Florida Artif. Intell. Res. Soc. Conf.*, 2017, pp. 353–358.

[18] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, and X. Cheng, “Text matching as image recognition,” in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2793–2799.

[19] Z. Wang, H. Mi, and A. Ittycheriah, “Sentence similarity learning by lexical decomposition and composition,” in *Proc. 26th Int. Conf. Comput. Linguistics: Tech. Papers*, 2016, pp. 1340–1349.

[20] B. Hu, Z. Lu, H. Li, and Q. Chen, “Convolutional neural network architectures for matching natural language sentences,” in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 2042–2050.

[21] W. Yin, H. Schütze, B. Xiang, and B. Zhou, “ABCNN: Attention-based convolutional neural network for modeling sentence pairs,” *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 259–272, 2016.

[22] W. Yin, S. Ebert, and H. Schütze, “Attention-based convolutional neural network for machine comprehension,” in *Proc. Human-Comput. Question Answering Workshop*, 2016, pp. 15–21.

[23] M. Korpusik and J. Glass, “Convolutional neural networks for dialogue state tracking without pre-trained word vectors or semantic dictionaries,” in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 884–891.

[24] M. Korpusik and Glass, “Convolutional neural encoder for the 7th dialogue system technology challenge,” in *Proc. 7th Dialogue Syst. Technol. Challenge Workshop*, 2019. [Online]. Available: http://workshop.colips.org/dstc7/workshop.html

[25] V. Mnih *et al.*, “Playing Atari with deep reinforcement learning,” in *Proc. Deep Learn. Workshop Conf. Neural Inf. Process. Syst.*, 2013, pp. 1–9.

[26] S. Young *et al.*, “The hidden information state model: A practical framework for POMDP-based spoken dialogue management,” *Comput. Speech Lang.*, vol. 24, no. 2, pp. 150–174, 2010.

[27] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky, “Deep reinforcement learning for dialogue generation,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1192–1202.

[28] K. Mo, S. Li, Y. Zhang, J. Li, and Q. Yang, “Personalizing a dialogue system with transfer learning,” 2016, arXiv:1610.02891.

[29] J. Li, A. Miller, S. Chopra, M. Ranzato, and J. Weston, “Learning through dialogue interactions,” 2016, arXiv:1612.04936.

[30] J. Henderson, O. Lemon, and K. Georgila, “Hybrid reinforcement/supervised learning for dialogue policies from communicator data,” in *Proc. IJCAI Workshop Knowl. Reasoning Practical Dialogue Syst.*, 2005, pp. 68–75.

[31] T. Zhao and M. Eskenazi, “Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning,” in *Proc. 17th Annu. Meeting Special Interest Group Discourse Dialogue*, 2016, pp. 1–10.

[32] B. Dhingra *et al.*, “End-to-end reinforcement learning of dialogue agents for information access,” 2016, arXiv:1609.00777.

[33] J. Williams and G. Zweig, “End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning,” 2016, arXiv:1606.01269.

[34] B. Peng *et al.*, “Composite task-completion dialogue system via hierarchical deep reinforcement learning,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2231–2240.

[35] X. Li, Y. Chen, L. Li, and J. Gao, “End-to-end task-completion neural dialogue systems,” 2017, arXiv:1703.01008.

[36] M. Korpusik, N. Schmidt, J. Drexler, S. Cyphers, and J. Glass, “Data collection and language understanding of food descriptions,” in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2014, pp. 560–565.

[37] M. Korpusik, C. Huang, M. Price, and J. Glass, “Distributional semantics for understanding spoken meal descriptions,” in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, 2016, pp. 6070–6074.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[39] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015, arXiv:1502.03167.

[40] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[41] M. Korpusik and J. Glass, “Spoken language understanding for a nutrition dialogue system,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 7, pp. 1450–1461, Jul. 2017.

[42] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Proc. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[43] M. Korpusik and J. Glass, “Convolutional neural networks and multitask strategies for semantic mapping of natural language input to a structured database,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 6174–6178.

[44] M. Korpusik, Z. Collins, and J. Glass, “Character-based embedding models and reranking strategies for understanding natural language meal descriptions,” *Proc. Interspeech*, 2017, pp. 3320–3324.

[45] S. Varges, S. Quarteroni, G. Riccardi, and A. Ivanov, “Investigating clarification strategies in a hybrid POMDP dialog manager,” in *Proc. 11th Annu. Meeting Special Interest Group Discourse Dialogue*, 2010, pp. 213–216.

Mandy Korpusik received the B.S. degree in electrical and computer engineering from Olin College of Engineering, Needham, MA, USA, in May 2013, and the S.M. degree in computer science from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in June, 2015. She is a Ph.D. candidate working with Dr. James Glass in the MIT Computer Science and Artificial Intelligence Laboratory, MIT. Currently, she is working on a spoken diet tracking system that uses convolutional neural networks to map between natural language meal descriptions and USDA food database entries. Her primary research interests include natural language processing and spoken language understanding for dialogue systems.

James Glass received the S.M. and Ph.D. degrees in electrical engineering and computer science from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He is a Senior Research Scientist with MIT, where he leads the Spoken Language Systems Group in the Computer Science and Artificial Intelligence Laboratory. His research interests include automatic speech recognition, unsupervised speech processing, and spoken language understanding. Dr. Glass is a member of the Harvard-MIT Health Sciences and Technology Faculty. He is a Fellow of the International Speech Communication Association, and is currently an Associate Editor for *Computer, Speech, and Language*.