Cooking with Semantics

Jon Malmaud, Earl J. Wagner, Nancy Chang, Kevin Murphy
malmaud@mit.edu, {wag,ncchang,kpmurphy}@google.com

Abstract

We are interested in the automatic interpretation of how-to instructions, such as cooking recipes, into semantic representations that can facilitate sophisticated question answering. Recent work has shown impressive results on semantic parsing of instructions with minimal supervision, but such techniques cannot handle much of the situated and ambiguous language used in instructions found on the web. In this paper, we suggest how to extend such methods using a model of pragmatics, based on a rich representation of world state.

1 Introduction

Understanding instructional text found on the web presents unique challenges and opportunities that represent a frontier for semantic parsing. Crucially, instructional language is *situated*: it assumes a situational context within which the agent (i.e., the reader) is to carry out a sequence of actions, as applied to objects that are (or become) available in the immediate environment. These actions and objects may not be explicitly specified; indeed, much instructional language is ambiguous, underspecified and often even ungrammatical relative to conventional usage.

In this "vision paper", we focus on interpreting cooking recipes. While there are several services that already support searching for recipes (such as Google Recipe Search[1], Yummly, Foodily, and MyTaste), the faceted search capabilities they provide are limited to recipe meta-data such as ingredients, genres, cooking time, portions, and nutrition values. Some of this information is explicitly marked up in machine-readable form[2]. However, the actual steps of the recipe are treated as an unstructured blob of text. (The same problem applies to other instructional sites, such as ehow.com, wikihow.com, answers.yahoo.com, www.instructables.com, etc.) Interpreting the steps of recipes (and instructions more generally) is the goal of this paper.

1. Sidecar recipe
A. Coat the rim of a cocktail glass with sugar and set aside.
B. Add the remaining ingredients to a shaker and fill with ice.
C. Shake and strain into the prepared glass.
D. Garnish with a piece of orange peel.

2. Guacamole recipe (first few steps)
A. Cut open avocados and scoop them out.
B. Discard skin and pits.
C. Use a fork or mixer to blend the avocados until smooth
D. ...

Figure 1: Example recipes. Left: for a mixed drink. Right: for guacamole dip.

2 Challenges

This section surveys some of the linguistic challenges typical of the cooking domain, as illustrated by the two recipes in Figure 1. These difficulties can be classified broadly as problems arising from the interpretation of **arguments**, **actions** and **control structure**.

Arguments: One particularly salient characteristic of recipes is that they often feature arguments that are omitted, underspecified or otherwise dependent on the context. Arguments may be *elided* in syntactic contexts where they are usually required (the so-called "zero anaphora" problem), especially when they are easily filled by an object in the immediate context. For example, the item to set aside in (1a) is the just-treated cocktail glass, and the item to fill in (1b) and shake and then strain in (1c) is the recently mentioned shaker. Note that the context may include the ingredient list itself, as illustrated by the elided argument(s) to be added in the one-line recipe "Add to a cocktail glass in the order listed." Arguments may be *implicitly available*, based on either domain-specific expectations of the initial context or the results of pre-

[1] http://www.google.com/insidesearch/features/recipes/

[2] See e.g. http://microformats.org/wiki/recipe-formats

ceding steps. The ice in (1b) isn't listed in the corresponding recipes ingredient list, since many common ingredients (water, ice, salt, pepper) are assumed to be available in most kitchens. Sometimes, the argument may never have been directly verbalized, but rather is the result of a previous action. Thus in the recipe "Pour ingredients over ice and shake vigorously," the object to shake is the container (only implicitly available) along with its contents — which, once the "pour" instruction is executed, include both ice and the (listed) ingredients. Note also that interpreting "the remaining ingredients" in (1b) requires an understanding of which ingredients have yet to be used at that point in the recipe. Arguments may be *indirectly available*, by association with an explicitly available argument. Recipe 2 mentions avocados in several explicit and implicit referring expressions; of these only the "them" in (2a) may be considered straightforward anaphoric reference (to the just-cut avocados). Step (2b) involves a metonymic reference to the "skin and pits" where the part-whole relation between these items and the avocado is what makes the instruction interpretable. Step (2c) once again mentions "avocados", but note that this now refers to the flesh of the avocados, i.e., the implicit scooped-out object from (2a). Arguments may be *incompletely specified*, especially with respect to amount. The exact amount of sugar needed in (1a) is not mentioned, for example. Similarly, the amount of ice needed in (1b) depends on the size of the shaker and is not precisely specified.

Actions: Like arguments, action interpretation also depends on the situational context. For example, actions may have *ambiguous senses*, mainly due to the elided arguments noted above. The verb "shake" in (1c), for example, yields a spurious intransitive reading. Actions may have *argument-dependent senses*: certain verbs may resolve to different motor actions depending on the affordances of their arguments. For example, the action intended by the verb "garnish" in (1d) might involve careful perching of the peel on the rim of the glass; in other recipes, the same verb applied to nutmeg or cut fruit may be better interpreted as an add action. Actions may be *omitted* or *implied*, in particular by the way certain arguments are expressed. Most recipes involving eggs, for example, do not explicitly mention the need to crack them and extract their contents; this is a default preparatory step. Other ingredients vary in how strongly they are associated with (implicit) preparatory steps. For example, recipes calling for "1/4 avocado" may require that something like steps (2a-b) be undertaken (and their results quartered); the "orange peel" of (1d) may likewise depend on a separate procedure for extracting peel from an orange.

Control structure: Instructions sometimes provide more complex information about sequence, coordination and control conditions. **Conditions**: An action may be specified as being performed until some finish condition holds. In (2c), the "until smooth" condition—itself featuring an elided avocado argument—controls how long the blending action should continue. Other conditions mentioned in recipes include "Add crushed ice until the glass is almost full", "Stir until the glass begins to frost", and "Add salt to taste". **Sequence**: Though implicitly sequential, recipes occasionally include explicit sequencing language. In the recipe "Add to a cocktail glass in the order listed", the order reflects that of the ingredient list. Other recipes specify that certain steps can or should be done "ahead of time", or else while other steps are in progress. **Alternatives**: Recipes sometimes allow for some variability, by specifying alternative options for specific ingredients ("Garnish with a twist of lemon or lime"), appliances or utensils ("Using a large fork (or a blender)..."), and even actions ("Chop or mash the avocados").

As should be clear from these examples, the interpretation of a given step in a set of instructions may hinge on many aspects of situated and procedural knowledge, including at least: the physical context (including the particular props and tools assumed available); the incremental state resulting from successful execution of previous steps; and general commonsense knowledge about the affordances of specific objects or expected arguments of specific actions (or more conveniently, corpus-based verb-argument expectations that approximate such knowledge, see e.g., (Nyga and Beetz, 2012)). All of these sources of knowledge go significantly beyond those employed in semantic parsing models for single utterances and in non-procedural contexts.

3 Proposed approach

We propose to maintain a rich latent context that persists while parsing an entire recipe, in contrast

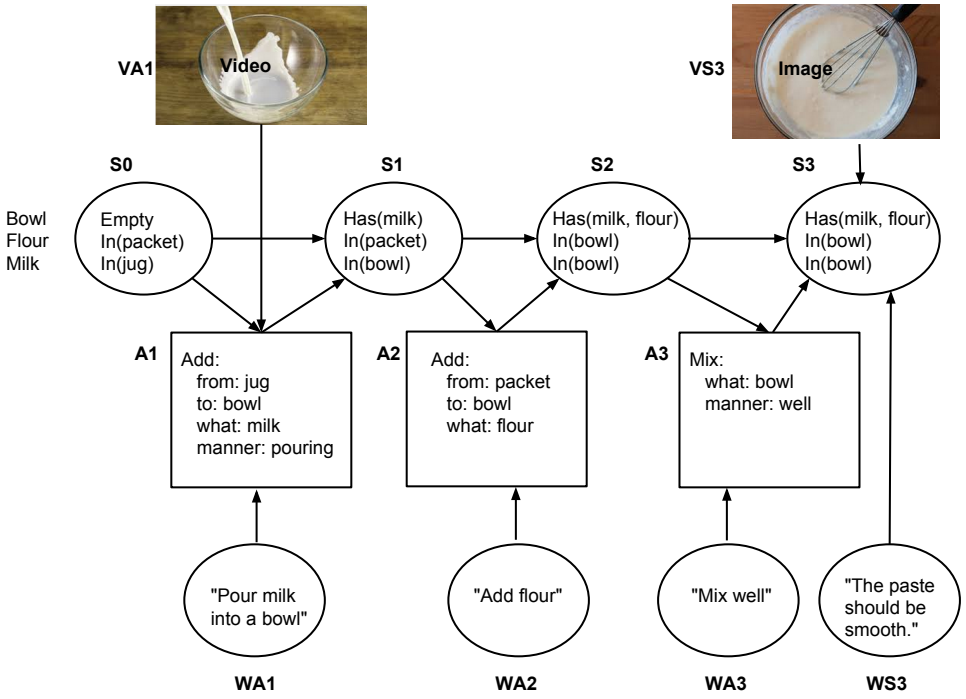


Figure 2: Our proposed probabilistic model, showing a possible trace of observed and latent variables after parsing each step of a pancake recipe. See text for description of notation.

to approaches that interpret each sentence independently. This context represents the state of the kitchen, and statements in the recipes are interpreted pragmatically with respect to the evolving context. More precisely, our model has the overall structure of a discrete-time, partially observed, object-oriented Markov Decision Process, as illustrated in Figure 2. The states and actions are both hidden. What we observe is text and/or images/video; our goal is to infer the posterior over the sequence of actions (i.e., to recover the "true" recipe), given the noisy evidence.

States and actions. The world state S_t is represented as a set of objects, such as ingredients and containers, along with various predicates, encoding the quantity, location, and condition (e.g., raw or cooked, empty or full) of each object. Note that previous work on situated semantic parsing often uses grid world environments where the only fluent is the agent's location; in contrast, we allow any object to undergo state transformations. In particular, objects can be created and destroyed.

Each action A_t is represented by a *semantic frame*, corresponding to a verb with various arguments or roles. This specifies how to transform the state. We also allow for sequencing and loop frames c.f., the "robot control language" in (Matuszek et al., 2013). We assume access to a simple cooking simulator that can take in a stream of low-level instructions to produce a new state; this implements the world dynamics model $p(\mathrm{S}_t|\mathrm{S}_{t-1}, \mathrm{A}_t)$.

Text data. We assume that the text of the t'th sentence, represented by WA_t, describes the t'th primitive action, A_t. We represent the conditional distribution $p(\mathrm{A}_t|\mathrm{WA}_t, \mathrm{S}_{t-1})$ as a log-linear model, as in prior work on frame-semantic parsing/ semantic role labeling (SRL) (Das et al., 2014).[3] However, we extend this prior work by allowing roles to be filled not just from spans from the text, but also by objects in the latent state vector. We will use various pragmatically-inspired features to represent the compatibility between candidate objects in the state vector and roles in the action frame, including: whether the object has been recently mentioned or touched, whether the object has the right affordances for the corresponding role (e.g., if the frame is "mix", and the role is "what", the object should be mixable),

[3] Although CCGs have been used in previous work on (situated) semantic parsing, such as (Artzi and Zettlemoyer, 2013), we chose to use the simpler approach based on frames because the nature of the language that occurs in recipes is sufficiently simple (there are very few complex nested clauses).

etc. More sophisticated models, based on modeling the belief state of the listener (e.g., (Goodman and Stuhlmüller, 2013; Vogel et al., 2013)) are also possible and within the scope of future work.

In addition to imperative sentences, we sometimes encounter descriptive sentences that describe what the state should look like at a given step (c.f., (Lau et al., 2009)). We let WS_t denote a sentence (possibly empty) describing the t'th state, S_t. The distribution $p(\mathrm{S}_t|\mathrm{WS}_t)$ is a discriminative probabilistic classifier of some form.

Visual data. Much instructional information is available in the form of how-to videos. In addition, some textual instructions are accompanied by static images. We would like to extend the model to exploit such data, when available.

Let a video clip associated with an action at time t be denoted by VA_t. We propose to learn $p(\mathrm{A}_t|\mathrm{VA}_t)$ using supervised machine learning. For features, we could use the output of standard object detectors and their temporal trajectories, as in (Yu and Siskind, 2013), bags of visual words derived from temporal HOG descriptors as in (Das et al., 2013), or features derived from RGB-D sensors such as Kinect, as in (Song et al., 2013; Lei et al., 2012).

There are many possible ways to fuse the information from vision and text, i.e., to compute $p(\mathrm{A}_t|\mathrm{VA}_t, \mathrm{WA}_t, \mathrm{S}_{t-1})$. The simplest approach is to separately train the two conditionals, $p(\mathrm{A}_t|\mathrm{WA}_t, \mathrm{S}_{t-1})$ and $p(\mathrm{A}_t|\mathrm{VA}_t)$, and then train another model to combine them, using a separate validation set; this will learn the relative reliability of the two sources of signal.

Learning and inference. We assume that we have manually labeled the actions A_t, and that the initial state S_0 is fully observed (e.g., a list of ingredients, with all containers empty). If we additional assume that the world dynamics model is known[4] and deterministic, then we can uniquely infer the sequence of states $\mathrm{S}_{1:T}$. This lets us use standard supervised learning to fit the log-linear model $p(\mathrm{A}_t|\mathrm{WA}_t, \mathrm{S}_{t-1})$.

In the future, we plan to relax the assumption of fully labeled training data, and to allow for learning from a distant supervision signal, similar to (Artzi and Zettlemoyer, 2013; Branavan et al., 2009). For example, we can prefer a parse that results in a final state in which all the ingredients have been consumed, and the meal is prepared.

[4]There has been prior work on learning world models from text, see e.g., (Sil and Yates, 2011; Branavan et al., 2012).

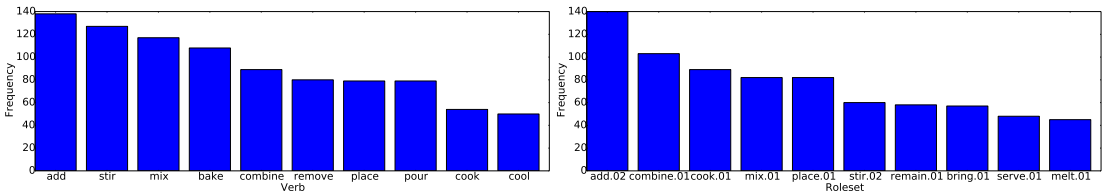
4 Preliminary results

We conducted a preliminary analysis to gauge the feasibility and expected performance benefits of our approach. We used the raw recipes provided in the CMU Recipe Database (Tasse and Smith, 2008), which consists of 260 English recipes downloaded from `allrecipes.com`. We then applied a state-of-the art SRL system (Das et al., 2014) to the corpus, using Propbank (Palmer et al., 2005) as our frame repository. Figure 3 summarizes our findings.

To judge the variance of predicates used in the cooking domain, we computed the frequency of each word tagged as a present-tense verb by a statistical part-of-speech tagger, filtering out a small number of common auxiliary verbs. Our findings suggest a relatively small number of verbs account for a large percentage of observed instructions (e.g, "add", "bake", and "stir"). The majority of these verbs have corresponding framesets that are usually correctly recognized, with some notable exceptions. Further, the most common observed framesets have a straightforward mapping to our set of kitchen state transformations, such as object creation via combination ("add", "mix", "combine", "stir in"), location transfers ("place", "set"), and discrete state changes over a small space of features ("cook", "cut", "cool", "bake").

To gain a preliminary understand of the limitations of the current SRL system and the possible performance benefits of our proposed system, we hand-annotated five of our recipes as follows: Each verb in the recipe corresponding to an action was annotated with its best corresponding roleset (if any). Each role in that roleset was marked as either being explicitly present in the text, implicitly present in our latent kitchen model but not in the text (and so in principle, fillable by our model), or neither present in the text nor in our model. For example, in"cover for forty minutes", the frameset "cover" has an explicit temporal role-filling ("for forty minutes") and an implicit role-filling ("the pot" as the patient "cover").

For each verb in the annotation, we checked if the SRL system mapped that verb to the correct roleset and if so, whether it filled the same semantic roles as the annotator indicated were explicitly present in the text. Overall, we found 54% recall of the annotations by the SRL system. We quali-



Heat oil in a large pot [until hot]; brown chicken [in the pot].
Remove chicken [from the pot] and set [the chicken] aside.
Saute onions until [the onions are] soft, about 5 minutes.
Add broth, beans, half the pepper, and all the chicken [to the pot]; cover and simmer [the pot contents] for 40 minutes.
Add parsley, cilantro, salt, and remaining pepper [to the pot], and simmer [the mixture] 5 more minutes.

Figure 3: Results. Top: Distribution of the ten most common verbs and framesets in 260 recipes from allrecipes.com. Bottom: An example recipe annotation. Blue indicates propbank predicates. Bracketed red indicates implicit propbank arguments not in the text, but in principle recognizable by our model. Green indicates quantifier adjectives which our model could resolve to an exact quantity, given initial ingredient amounts.

tatively notes several failure modes. Many errors arise from not recognizing predicates represented in the text as an imperative verb, likely because PropBank contains few examples of such language for the labeler to learn from. Other errors result from ungrammatical constructs (e.g. in "cook five minutes", the eliding of "for" causes "five minutes" to incorrectly parse as a direct argument). Certain cooking-related verbs lack framesets entirely, such as "prebake". Occasionally, the wrong roleset is chosen. For example, in"Stir the mixture" , "Stir" is labeled as "stir.02: cause (emotional) reaction" rather than "stir.01: mix with a circular motion".

We also analyzed the quantity and qualitative trends in the human annotations that refer to roles fillable from the latent kitchen model but not literally present in the text. Overall, 52% of verb annotations referenced at least one such role. The most common situation (occurring for 36% of all annotated verbs) is the "patient/direct object" role is elided in the text but inferable from the world state, as in "simmer [the mixture] for 40 minutes". The second most common is the "location" modifier role is elided in the text, as in "Remove chicken [from the pot]". Overall, we believe our proposed approach will improve the quality of the SRL system, and thus the overall interpretability of the recipes.

5 Possible applications

We believe that semantic parsing of recipes and other instructional text could support a rich array of applications, such as the following:

Deriving a "canonical" recipe. It would be useful to align different versions of the same recipe to derive a "canonical form" cf., (Druck and Pang, 2012; Tenorth et al., 2013b).

Explaining individual steps. It would be helpful if a user could click on a confusing step in a recipe and get a more detailed explanation and/or an illustrative video clip.

Automatically interpreting software instructions. Going beyond the recipe domain, it would be useful to develop a system which can interpret instructions such as how to install software, and then automatically execute them (i.e., install the software for you). In practice, this may be too hard, so we could allow the system to ask for human help if it gets stuck, cf. (Deits et al., 2013).

Robotics. (Tenorth et al., 2013a) suggest mining natural language "action recipes" as a way to specify tasks for service robots. In the domain of food recipes, there have already been several demonstrations (e.g., (Beetz et al., 2011; Bollini et al., 2013)) of robots automatically cooking meals based on recipes.

Task assistance using augmented reality. Imagine tracking the user as they follow some instructions using a device such as Google glass, and offering help when needed. Such systems have been developed before for specialized domains like maintenance and repair of military hardware[5], but automatic parsing of natural language text potentially opens this up to the consumer market. (Note that there is already a recipe app for Google Glass[6], although it just displays a static list of instructions.)

[5]For example, see http://graphics.cs.columbia.edu/projects/armar/index.htm.

[6]See http://www.glassappsource.com/listing/all-the-cooks-recipes.

References

Y. Artzi and L. Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Trans. Assoc. for Computational Linguistics*, 1:49–62.

M. Beetz, U. Klank, I. Kreese, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth. 2011. Robotic roommates making pancakes. In *Intl. Conf. on Humanoid Robots*.

Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. 2013. Interpreting and executing recipes with a cooking robot. *Experimental Robotics*.

SRK Branavan, H. Chen, L. Zettlemoyer, and R. Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Association for Computational Linguistics*.

S.R.K. Branavan, N. Kushman, T. Lei, and R. Barzilay. 2012. Learning High-Level Planning from Text. In *ACL*.

P. Das, C. Xu, R. F. Doell, and J. J. Corso. 2013. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *CVPR*.

D. Das, D. Chen, A. Martins, N. Schneider, and N. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*.

R. Deits, S. Tellex, P. Thaker, D. Simeonov, T. Kollar, and N. Roy. 2013. Clarifying Commands with Information-Theoretic Human-Robot Dialog. *J. Human-Robot Interaction*.

G. Druck and B. Pang. 2012. Spice it Up? Mining Renements to Online Instructions from User Generated Content. In *ACL*.

Noah D Goodman and Andreas Stuhlmüller. 2013. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science*, 5(1):173–184.

TA Lau, Clemens Drews, and Jeffrey Nichols. 2009. Interpreting Written How-To Instructions. *IJCAI*.

J. Lei, X. Ren, and D. Fox. 2012. Fine-grained kitchen activity recognition using RGB-D. In *Ubicomp*.

C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox. 2013. Learning to parse natural language commands to a robot control system. *Experimental Robotics*, pages 1–14.

D. Nyga and M. Beetz. 2012. Everything robots always wanted to know about housework (but were afraid to ask). In *Intl. Conf. on Intelligent Robots and Systems*.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

A. Sil and A. Yates. 2011. Extracting STRIPS Representations of Actions and Events. In *Recent Advances in NLP*.

Young Chol Song, Henry Kautz, James Allen, Mary Swift, Yuncheng Li, Jiebo Luo, and Ce Zhang. 2013. A markov logic framework for recognizing complex events from multimodal data. In *Proc. 15th ACM Intl. Conf. Multimodal Interaction*, pages 141–148. ACM.

D. Tasse and N. Smith. 2008. SOUR CREAM: Toward Semantic Processing of Recipes. Technical Report CMU-LTI-08-005, Carnegie Mellon University, Pittsburgh, PA.

M. Tenorth, A. Perzylo, R. Lafrenz, and M. Beetz. 2013a. Representation and exchange of knowledge about actions, objects, and environments in the roboearth framework. *IEEE Trans. on Automation Science and Engineering*, 10(3):643–651.

M. Tenorth, J. Ziegltrum, and M. Beetz. 2013b. Automated alignment of specifications of everyday manipulation tasks. In *IEEE Intl. Conf. on Intelligent Robots and Systems*.

A. Vogel, M. Bodoia, C. Potts, and D. Jurafsky. 2013. Emergence of Gricean Maxims from Multi-Agent Decision Theory. In *NAACL*.

Haonan Yu and JM Siskind. 2013. Grounded language learning from video described with sentences. In *Association for Computational Linguistics*.