Recipe Attribute Prediction using Review Text as Supervision

Gregory Druck
Yummly, Inc.
greg@yummly.com

Abstract

Online recipes are often accompanied by user reviews. In addition to numeric ratings and descriptions of modifications, these reviews frequently contain detailed information about the cooking process, the taste and texture of the dish, and occasions or situations for which the dish is suited. In this paper, we aim to leverage this information to build a system that predicts what users *would* say about a recipe. Specifically, we annotate recipes with attributes that are applied to them in reviews. Then, we train models to predict these attributes using information about the ingredients, preparation steps, and recipe title. For example, we aim to predict whether a salad would be described as "refreshing" in reviews. We demonstrate that it is possible to make such predictions accurately and that the factors that are important in these predictions are intuitive. We also discuss potential downstream applications of this method to recipe recommendation, recipe retrieval, and guided recipe modification.

1 Introduction

There has been substantial interest in improving recipe retrieval and recommendation by augmenting recipe representations with various types of metadata, including course and dish tags, quality estimates, and possible modifications [Badra *et al.*, 2008; Teng *et al.*, 2011; Wang *et al.*, 2008; Zhang *et al.*, 2008]. In particular, prior work has shown that substitutions, omissions, and other recipe modifications can be gleaned from the text of user reviews [Teng *et al.*, 2011; Druck and Pang, 2012]. However, reviews also provide information about the taste of the dish (*"Great but next time I will use low sodium ham because it was WAY too salty."*), the texture (*"It is rich and textured, wonderfully creamy."*), how the dish was received (*"My kids loved this recipe because there weren't any strong flavors, it was all so well blended."*), and how the dish made the reviewer feel (*"What a comforting, savory dish!"*). In this paper, we leverage reviews to augment recipe representations with such information.

Unfortunately, appropriate reviews are not always available. Small recipe sites may lack reviewing functionality or a sizable community of reviewers. On sites with large numbers of recipes, a new recipe may never be reviewed because users are weary of trying it until someone else reviews it. Even when a recipe has several reviews, the particular aspect of interest may not be discussed, or there may be insufficient evidence to make the desired inference.

In this paper, we aim to predict what a review *would* say about a recipe. For example, we would like to predict how "creamy" a particular soup will be, or how "chewy" a particular batch of cookies will be, without using reviews. To do this, we identify applications of *recipe attributes* (e.g. *creamy*, *chewy*) in review text for a set of recipes (e.g. *soup* recipes, *cookie* recipes). Then, we train models that rank recipes by the likelihood that they have different attributes using only features of the recipe. This enables augmenting recipe representations when reviews are unavailable, and is a step toward a deeper (computational) understanding of recipes.

More specifically, we first select a set of recipes and accompanying reviews. The reviews are analyzed, and selected words and phrases from the reviews are proposed as candidate attributes. In this paper, we manually select a subset of the more interesting candidates. Then, we build rule-based taggers that identify applications of attributes to recipes in reviews. This part of our approach is discussed in Section 3.

Next, we aggregate applications of attributes across reviews to obtain an attribute probability for each recipe and attribute pair. The attribute probability is the proportion of reviews of a recipe that apply a particular attribute. This data is then used to train ranking models for each attribute. In particular, we take a pairwise learning to rank approach [Herbrich *et al.*, 2000; Joachims, 2002; Burges *et al.*, 2005]. This involves training models that, given a pair of recipes, predict the recipe with the higher attribute probability. Because the attribute probability estimates are noisy, we are careful to use pairs that we are confident have different attribute probabilities. The features of the ranking models include the recipe ingredients, abstractions of the ingredients, ingredient amounts, and features derived from the preparation steps and recipe title. This part of our approach is discussed in Section 4.

We compare a baseline and several variants of our approach on 15 prediction tasks. Our approach yields accurate predictions for these tasks and substantially outperforms the baseline. We conduct ablation experiments in order to determine the importance of different types of features. We also provide data that allows for qualitative evaluation of the trained mod-

els (Section 5.2). Finally, we discuss potential applications (Section 6) and provide discussion (Section 7).

2 Related Work

We are not aware of prior work that identifies recipe attributes such as taste, texture, and relevant occasions in user reviews, or prior work that trains models to predict attributes from a representation of the recipe. More generally, we are not aware of prior work that predicts review text using a description of the item being reviewed, or prior work that uses review text as a source of supervision.

Prior work in recipe review text analysis has focussed on extracting substitutions and other modifications of the recipe [Teng *et al.*, 2011; Druck and Pang, 2012]. In contrast, in this paper we focus on extracting other types of information from the reviews (e.g. taste, texture, occasion). The review analysis component of our approach (Section 3.3) could easily be adapted to identify modifications as well. This would enable the *prediction* of recipe modifications.

Prior work that uses recipe reviews as a source of supervision has focussed on utilizing review scores. In particular, Teng et al. [2011] estimate ingredient co-occurrence and substitution networks from recipes and reviews, and use features of these networks to predict review scores. In contrast, we aim to predict attributes derived from review text, rather than review scores. Additionally, our approach uses abstractions of ingredients, ingredient amounts, information about the preparation steps, and the recipe title to make predictions.

More generally, there has been interest in augmenting recipe representations with metadata or additional structure. Zhang et al. [2008] and Badra et al. [2008], among others, use case-based representations and case-based reasoning for recipe retrieval. Both works additionally augment recipe case representations with predicted attributes such as dishes, courses, and cuisines. Our method could be used for these prediction tasks, but supervision for them is more reliably available in other forms, as most online recipe sites include cuisine and course annotations. Instead, we focus on attributes for which reviews are the best or only source of supervision. Due to the nature of review-based supervision, we also use a learning to rank approach, rather than a classification approach. The attributes that our approach predicts could be used to further augment a case-based recipe representation.

There are several papers that study *recipe recommendation*, though each addresses a slightly different task. While Teng et al. [2011] predict a proxy for overall recipe popularity, other work addresses personalized recipe recommendation based on explicit ratings [Forbes and Zhu, 2011; Freyne *et al.*, 2011] or implicit interactions (e.g. recipe page views) [Ueda *et al.*, 2011]. The focus of our work is quite different. Our goal is to use information from reviews to enhance our representation and understanding of recipes, rather than recommend recipes to users. However, the attribute predictions our method produces could be used as an additional input to recommendation systems.

Other work aims to derive a structured representation of the recipe from its preparation steps. For example, Wang et al. [2008] represent the preparation steps as a graph in which nodes are steps and directed edges are dependencies between steps. They then identify common subgraphs and use them to develop a new recipe representation that facilitates similarity computation. In this paper, we use a bag-of-words representation of the preparation steps, but we plan to consider a structured representation in future work.

Finally, there is prior work that aims to predict review text from a partial review. Bridge and Healy [2012] develop GhostWriter-2.0, a system that assists users in writing product reviews. Given a partial review, the system suggests additional product-specific aspects that could be addressed. In contrast, we predict review text from the item being reviewed.

3 Identifying Recipe Attributes in Reviews

We now describe our approach. We refer to the properties of recipes we aim to predict as *recipe attributes*. Examples include:

- tastes/flavors: *spicy*, *bitter*, *rich*, *earthy*
- textures: *creamy*, *crunchy*, *chewy*, *hard*
- occasions: *party*, *picnic*, *Thanksgiving*, *summer*
- other: *kid friendly*, *comfort food*, *easy*, *refreshing*

In this section we discuss identifying recipe attributes in reviews. In Section 4, we discuss training models to predict recipe attributes using features of the recipe.

3.1 Filtering by Dish

We may be interested in predicting an attribute for a particular type of dish, e.g. chewy cookies (see also the discussion in Section 7). Therefore, the input to the system is a set of recipes with accompanying reviews, along with an optional dish filter. If the dish filter is invoked, only recipes for a certain dish will be retained in subsequent steps. Identifying the dish is a non-trivial problem. In this paper, we find that the following heuristic, based on the recipe title, is effective, despite its simplicity:

1. Strip trailing prepositional phrases like "with Basil" from the recipe title, as well as numbers ("Pumpkin Pie IV"), the word "Recipe", and other non-critical content like stylistic punctuation.
2. Extract as the dish name the last k words of the remaining recipe title. For example if the recipe name is "Roasted Carrot and Tomato Soup with Basil", for $k = 4$ (the value we use in this paper) we would extract "Carrot and Tomato Soup" as the dish.

A recipe is filtered if its dish name does not end with the supplied dish filter.

3.2 Selecting Attribute Types

In this paper, we manually select a set of interesting attributes from reviews with the aid of computational tools. Specifically, common phrases and bigrams that are not ingredients or function words are identified as candidates. A dish filter can also be specified. In this case, we require candidate words and phrases to be especially likely in reviews of the

dish[1]. For the purposes of this paper, we selected a diverse set of 15 attributes from the candidate lists that span the categories provided above. Of the 15, 8 have an associated dish filter. Table 1 displays the full set of attributes with dish filters. As we discuss in Section 7, we plan to avoid this manual selection step in future work.

3.3 Identifying Applications of Recipe Attributes in Reviews

The next step is to annotate reviews with the attributes they apply to the recipe. We assume each attribute is either *present* or *absent* in each review. Although some attributes like *moistness* would ideally be expressed numerically (how moist?), we leave the extraction of a numeric value from review text ("very moist"$\mapsto$?) for future work.

Teng et al. [2011] found that a rule-based approach was sufficient to identify substitutions, additions, and deletions in reviews. Similarly, we find that a simple rule-based approach is sufficient to identify attributes. In our method, reviews are first split into sentences. Each attribute has its own "tagger" with inclusion and exclusion regular expressions. The tagger returns *present* if at least one sentence matches the inclusion regular expression and does not match the exclusion regular expression. By default, the inclusion regular expression requires the attribute word or phrase to be present and the exclusion regular expression forbids negations like "this isn't easy." If required, the defaults can be overridden to incorporate synonyms, stemming, and special cases, though we did not find this to be necessary in this work.

Though this method is very simple, we find that it is surprisingly precise. Based on a sample of 100 reviews, the precision of the default tagger for "easy" is 97% and the default tagger for "moist" (with a *cake* dish filter) is 93%, where a correct prediction occurs when the reviewer is actually expressing that the recipe has the attribute. One common type of error occurs when the user describes their modification to the recipe, rather than the original. The simplicity and precision of this method likely comes at the expense of recall, but this concern is mitigated by the fact that we aggregate attribute predictions over a large number of reviews.

4 Predicting Recipe Attributes

We now have a set of reviews annotated with binary attributes. We next compute *attribute probabilities* for each recipe. The attribute probability for a particular recipe and attribute is the proportion of reviews of the recipe that apply the attribute. For example, if a particular recipe has 10 reviews, and the attribute *easy* is applied in 4 of them, the *easy* attribute probability is 0.4. Our approach involves training independent models to predict each attribute using recipes annotated with attribute probabilities. In the rest of this section we describe the procedure for a single attribute. We denote the set of recipes with attribute probabilities by $\mathcal{D} = \{(x^1, a^1), \ldots, (x^n, a^n)\}$, where x^i is the ith recipe, a^i is the attribute probability for the ith recipe, and n is the total number of recipes.

[1]We consider a word or phrase to be especially likely if its pointwise mutual information in reviews of the dish is ≥ 2.

4.1 Ranking Model and Estimation

We next learn to rank[2] recipes according to their attribute probabilities. See [Liu, 2009] for a survey of learning to rank (LTR) methods. The input to LTR methods is a set of training items with scores. The goal is to learn a model that can produce a similar ranking given data with unknown scores — in our case recipes that have few or no reviews. The scores in our setting are attribute probabilities. Rather than defining a model that evaluates an entire ranking at once, which presents computational challenges because there are an exponential number of possible orderings, we use a pairwise decomposition of the ranking problem. In pairwise LTR [Herbrich *et al.*, 2000; Joachims, 2002; Burges *et al.*, 2005], the model evaluates the ordering of pairs of items. Training is performed by generating pairs of recipes from the training list and encouraging the model to predict the recipe with the larger attribute probability. To rank recipes with unknown attribute probabilities, the recipes are scored using the trained model and sorted. In this paper we evaluate rankings of recipes, but one could obtain classifications from the ranked list by specifying a threshold on the score.

Because the attribute probabilities are noisy estimates of the applicability of an attribute to a recipe, we only generate training pairs when we are fairly confident that the recipes have different attribute probabilities. Otherwise, the model may be overwhelmed by noise, or may waste effort trying to swap the ordering of pairs that are incorrectly labeled. To avoid this, we estimate a confidence interval around each attribute probability a^i, and only generate pairs for which 80% confidence intervals do not overlap[3]. The set of unique recipe pairs from $\mathcal{D}$ that satisfy this constraint is denoted by $\mathcal{P}$.

We use logistic regression models for pairwise prediction. Each recipe x^i is represented by a feature vector of dimensionality d. The value of the kth feature for recipe i is denoted by x^i_k. We discuss the features we use in Section 4.2. The pairwise labels y^{ij} are $\in \mathcal{Y} = \{0, 1\}$, where $y^{ij} = 1$ if $a^i > a^j$ and $y^{ij} = 0$ otherwise. That is, the label y^{ij} has value 1 when recipe x^i has the larger attribute probability. The probability of this event is given by

$$p(y^{ij}=1|x^i, x^j; \theta) = \frac{1}{1 + \exp\Big(-\sum_{k=1}^{d} \theta_k (x^i_k - x^j_k)\Big)},$$

We estimate model parameters θ by maximizing the log-likelihood of the training pairs and a Laplace prior on parameters (i.e. L_1 regularization). The objective function is

$$\mathcal{L}(\theta) = \sum_{(i,j)\in\mathcal{P}} \log p(y^{ij}|x^i, x^j; \theta) - \beta \sum_{k=1}^{d} |\theta_k|. \quad (1)$$

It is well-known that this is a convex function. We optimize $\mathcal{L}(\theta)$ using the Orthant-Wise Limited-memory Quasi-Newton

[2]In an earlier version of this work, we attempted to predict attribute probabilities directly using regression models. However, we found that the ranking approach yielded more accurate predictions.

[3]We use an 80% confidence interval because there is a tradeoff between having a large amount of noisy data and a small amount of very clean data. An 80% interval filters out many pairs but retains enough to allow the training of an accurate model.

method [Andrew and Gao, 2007]. The regularization parameter β is set to 1000 in all experiments. This constant is selected to encourage aggressive regularization in order to compensate for noise in the data and to produce sparse models (a property of L_1 regularization) that are easier to interpret.

4.2 Recipe Representation and Features

For the purposes of this paper, a recipe consists of a *title*, a set of *ingredients*, and an ordered list of *preparation steps*. In this section we discuss the features of this representation that are used to make predictions.

We first discuss ingredient features. Our approach leverages two base components: an ingredient ontology and a system for extracting information from ingredient lines. We do not discuss these components in detail in this paper. Importantly, the ingredient ontology allows abstraction of ingredients through is-a relationships. For example, the ontology encodes that *cauliflower* is a type of *inflorescent vegetable*. The ingredient line analyzer extracts the ingredient, amount, and unit from an ingredient line. For example, the ingredient line analyzer extracts {*amount* : 0.5, *unit* : cup, *ingredient* : cauliflower} from "1/2 cup cauliflower, cut into fine shreds."

Ingredient features consist of 1) binary features that represent the presence of an ingredient in a recipe, 2) binary features that represent the presence of an abstraction of an ingredient in a recipe (using the ontology), 3) numeric features that represent the amount of an ingredient in a recipe, and 4) binary features that represent a discretized version of the amount of an ingredient in a recipe.

The ingredient amounts are not very usable in their original form. We perform three steps to address this. First, we convert the amount of each ingredient to mass using data from the United States Department of Agriculture (USDA)[4]. If we cannot perform this conversion, the ingredient is ignored. Knowing the mass of a particular ingredient is not necessarily useful because different recipes produce different quantities. To compensate for this, we normalize each ingredient's mass by the total mass of all ingredients in the recipe. Finally, different ingredients may have very different mean values. For example, broth often makes up a large proportion of a soup, whereas salt and spices typically make up a very small proportion. To compensate for this, we *standardize* the amount features using

$$x'_k = \frac{x_k - \bar{x}_k}{\sigma_k},$$

where $\bar{x}_k$ is the mean feature value and σ_k is the feature standard deviation (both computed using all recipes in $\mathcal{D}$). The resulting feature values can then be interpreted as the number of standard deviations above or below the mean amount $\bar{x}_k$. Finally, the binary ingredient amount features have value 1 if the amount (number of standard deviations above or below the mean) is > 1, > 2, < -1, or < -2. We refer to these features as *bin features*.

For preparation steps, we first extract preparation methods, cooking methods, and equipment using dictionaries obtained from Wikipedia[5]. We then add binary features that indicate whether particular methods or types of equipment appeared in any preparation step.

Finally, we use features of the recipe title. The title often provides a tremendous amount of information, including the name of the dish, the main ingredients, and words that describe the recipe. In this paper, we extract binary unigram word features from the title, ignoring punctuation and case.

5 Experiments

We downloaded 4.2M reviews from four major recipe sites: food.com, allrecipes.com, epicurious.com, and foodnetwork.com. We remove recipes that have less than 10 reviews. After filtering, there are 67,512 unique recipes, which we process as described in Section 4.2.

Table 1 displays the list of recipe attributes we use in the experiments, the accompanying dish filters, and excerpts of reviews that apply the attributes to a recipe. We refer to each attribute-dish pair as a *task*.

Note that we use noisy data derived from review text for both training and evaluation. As a result, we perform both quantitative and qualitative evaluation of the trained models. See Section 7 for additional discussion of the advantages and disadvantages of deriving supervision from reviews.

5.1 Quantitative Evaluation

We compare our approach using all features with a simple *baseline* that ranks recipes that have the attribute name in the recipe title above those that do not. We also compare different versions of the learning to rank feature set. In particular, we conduct an ablation study where one set of features is removed in each trial. We perform one trial for each of the following feature sets. The *ingredients* feature set includes all ingredient features. The *abstractions* feature set includes all abstractions of the original ingredients. The *amounts* feature set includes all normalized ingredient amount features (for both the original and abstracted ingredients). The *preparation* feature set includes all preparation step features. Finally, the *title* feature set includes all title features. To increase efficiency and discourage overfitting, we prune features that occur in fewer than three recipes during feature processing.

We compare methods by measuring pairwise accuracy and NDCG on held-out test data. Pairwise accuracy is the percentage of pairs for which a method predicts the correct ordering. Note that this only includes pairs with significantly different attribute probabilities, as described in Section 4.1. Random guessing would yield pairwise accuracy of 50%.

Discounted cumulative gain (DCG) is a popular metric in the information retrieval literature for evaluating the quality of a ranking [Järvelin and Kekäläinen, 2002].

$$DCG_p = a^{r(1)} + \sum_{i=2}^{p} \frac{a^{r(i)}}{\log_2 i}, \quad (2)$$

In Equation 2, $r(i)$ is the index of the ith ranked recipe and $a^{r(i)}$ is the attribute probability for the ith ranked recipe.

[4]For example, see `http://reedir.arsnet.usda.gov/codesearchwebapp/codesearch.aspx`.

[5]For example, see `http://en.wikipedia.org/wiki/Category:Cooking_techniques`.

attribute	dish filter	review excerpt
comfort food	—	*"Deep and rich and perfect comfort food."*
kids loved	—	*"The kids loved the creamsicle flavor."*
party	—	*"I made this recipe for a superbowl party."*
picnic	—	*"A perfect pot luck or picnic salad!"*
winter	—	*"I'll definitely be making this again this winter."*
easy	—	*"This is just too easy to make, I love it!"*
spicy	—	*"great flavors, nice and spicy, MMMmmmm good."*
fell apart	burgers	*"These tested great but fell apart in cooking."*
dry	cake	*"This cake had a good taste but found it to be a little dry."*
moist	cake	*"I followed the recipe exactly and the cake came out moist and delicious."*
chewy	cookies	*"Like gingersnaps but soft and chewy."*
colorful	salad	*"The variety of veggies is very colorful."*
refreshing	salad	*"Nothing special, but it was tasty and refreshing."*
bland	soup	*"But even getting passed [sic] that, the soup was pretty bland."*
creamy	soup	*"So nice and creamy without being tooooooo fatty."*

Table 1: The list of recipe attributes we use in the experiments, the accompanying dish filters, and excerpts of reviews that apply the attributes to a recipe.

NDCG_p is DCG_p normalized by the DCG_p value of the optimal ranking. In order to evaluate the complete ranking of recipes, we use $p = n$. The range of NDCG values varies based on the distribution of attribute probabilities, and hence the NDCG results are not directly comparable across different tasks. For each task we present mean values of pairwise accuracy and NDCG obtained using 10-fold cross validation.

Table 2 compares the baseline method and the proposed approach using the full feature set. The proposed approach significantly outperforms the baseline in all experiments (Wilcoxon signed-rank tests, $p < 0.01$.). The baseline performs best on the *creamy (soup)* task, as creamy soups sometimes have the word "creamy" in the title. But the proposed approach is able to identify many other creamy soups, and as a result yields much higher accuracy (0.846 vs. 0.559) and NDCG (0.680 vs. 0.513). Note that for some tasks, such as *bland (soup)*, the baseline pairwise accuracy is random (0.500). Recipe authors are very unlikely to put negative attributes in the titles of their recipes. The results provided by our approach demonstrate that it is possible to accurately rank recipes by attribute probability. In Section 5.2 we provide additional qualitative evaluation.

Table 3 displays the results of ablation experiments. The goal of these experiments is to determine the relative importance of the feature sets described above. The values are differences between the accuracy/NDCG obtained using all features (displayed in Table 2) and the accuracy/NDCG obtained when one set of features is excluded. Statistically significant differences are displayed in bold (Wilcoxon signed-rank test, $p < 0.05$). Note that a significant decrease implies that the excluded feature set is important for the task. The last row displays the number of experiments in which excluding the feature set significantly reduces accuracy/NDCG. From these results, we conclude that title and ingredient features are the most important (15 and 14 significant decreases, respectively). Ingredient abstractions are often important as well (11 significant decreases). Ingredient amount and preparation step features are considerably less important (4 and 2 significant decreases). This may be an indication that we need more complex amount and preparation step features, as intuitively we would expect such information to be very useful.

Though often significant, many of the absolute differences are quite small. This suggests that although some feature sets are more important than others, the model is mostly able to compensate for the exclusion of a feature set using other features. Note that there is redundancy among title features, ingredient features, and preparation features, as demonstrated by a recipe titled "Baked Chicken with Roasted Tomatoes." Additionally, we find that excluding a feature set only significantly increases performance in one case. This suggests that using all features is rarely harmful, despite the redundancy.

In some cases, the removal of a feature set affects pairwise accuracy in a different way than NDCG. This occurs because the metrics measure different things. Pairwise accuracy emphasizes correctly ordering a pair of recipes, no matter where those recipes fall in the true ranking. In contrast, NDCG puts more emphasis on the ordering of the recipes with the largest attribute probabilities.

5.2 Qualitative Evaluation

In this section, we provide qualitative evaluation of the trained models by displaying important features, which provides intuition about how the predictions are made, and supplying example predictions on held-out recipes.

Running ablation tests that exclude each feature individually would be prohibitively expensive, and given the small differences observed in Table 3, it would be unlikely to produce meaningful results. Consequently, Table 4 displays the features with the maximum and minimum weights, subject to certain criteria, for six of the tasks. In particular, we require that the confidence intervals for these estimates do not contain 0, and that the corresponding feature be observed in at least 20 recipes. Features with weights > 0 are positive indicators that increase the scores of recipes that have them. Features with weights < 0 are negative indicators. The abbreviations *ingr*, *amt*, and *prep* stand for ingredients, amount, and prepa-

	no ingredients		no abstractions		no amounts		no preparation		no title	
attribute (dish)	Δacc	Δndcg	Δacc	Δndcg	Δacc	Δndcg	Δacc	Δndcg	Δacc	Δndcg
comfort food	-0.015	-0.022	-0.015	-0.022	-0.003	-0.013	-0.006	-0.011	-0.005	-0.013
easy	**-0.047**	-0.011	**-0.047**	-0.011	**-0.012**	-0.003	**-0.011**	+0.000	**-0.037**	**-0.013**
kids loved	**-0.033**	**-0.021**	**-0.033**	**-0.021**	-0.010	-0.012	-0.006	-0.008	-0.006	**-0.013**
party	-0.015	-0.013	-0.015	-0.013	**-0.013**	-0.028	-0.004	+0.004	**-0.022**	**-0.048**
picnic	**-0.061**	+0.003	**-0.061**	+0.003	+0.004	+0.000	-0.011	-0.012	-0.025	-0.011
spicy	**-0.059**	-0.050	**-0.059**	-0.050	-0.002	+0.000	+0.003	+0.002	**-0.014**	**-0.051**
winter	**-0.026**	**-0.035**	**-0.026**	**-0.035**	-0.010	+0.001	-0.005	-0.002	**-0.023**	-0.020
fell apart (burgers)	**-0.163**	**-0.140**	**-0.169**	**-0.141**	+0.000	+0.000	-0.004	+0.016	-0.005	-0.001
dry (cake)	**-0.039**	-0.016	+0.005	**+0.016**	-0.002	+0.012	-0.005	+0.020	-0.006	+0.003
moist (cake)	**-0.034**	-0.012	-0.005	-0.003	**-0.012**	+0.001	-0.001	+0.006	-0.010	-0.010
chewy (cookies)	**-0.034**	-0.012	-0.011	-0.002	**-0.010**	-0.007	+0.000	-0.003	**-0.015**	**-0.084**
colorful (salad)	-0.028	+0.006	-0.011	+0.005	-0.004	+0.000	+0.014	+0.004	**-0.020**	+0.013
refreshing (salad)	-0.010	+0.011	-0.009	-0.002	-0.001	+0.006	+0.001	+0.006	**-0.016**	**-0.035**
bland (soup)	**-0.029**	-0.010	-0.011	-0.012	-0.002	-0.005	**-0.023**	+0.000	-0.006	-0.002
creamy (soup)	**-0.015**	+0.009	**-0.015**	+0.009	+0.003	-0.002	+0.001	-0.005	**-0.015**	**-0.032**
significant decreases	11	3	7	4	4	0	2	0	8	7

Table 3: Ablation study. The values are differences between the accuracy/NDCG obtained using all features and the accuracy/NDCG obtained when one set of features is excluded. Statistically significant differences are displayed in bold (Wilcoxon signed-rank test, $p < 0.05$). The last row displays the number of experiments in which excluding the feature set significantly reduces accuracy/NDCG. Note that a significant decrease implies that the excluded feature set is important for the task. We see that title, ingredient, and ingredient abstraction features are more important than amount and preparation step features.

	baseline		proposed method	
attribute (dish)	acc	NDCG	acc	NDCG
comfort food	0.502	0.321	**0.896**	**0.499**
easy	0.520	0.784	**0.716**	**0.856**
kids loved	0.500	0.404	**0.796**	**0.472**
party	0.505	0.537	**0.733**	**0.697**
picnic	0.501	0.280	**0.830**	**0.454**
spicy	0.555	0.434	**0.873**	**0.692**
winter	0.515	0.368	**0.884**	**0.575**
fell apart (burgers)	0.477	0.461	**0.678**	**0.635**
dry (cake)	0.500	0.550	**0.733**	**0.653**
moist (cake)	0.510	0.747	**0.825**	**0.860**
chewy (cookies)	0.549	0.610	**0.824**	**0.814**
colorful (salad)	0.495	0.392	**0.720**	**0.448**
refreshing (salad)	0.503	0.535	**0.764**	**0.710**
bland (soup)	0.500	0.534	**0.677**	**0.609**
creamy (soup)	0.559	0.513	**0.846**	**0.680**

Table 2: Comparison of the baseline and the proposed method (using all features). The proposed method significantly outperforms the baseline on all attribute prediction tasks (Wilcoxon signed-rank tests, $p < 0.01$.).

ration steps, respectively. Ingredient amount features that are followed by $(\pm c\sigma)$ are bin features (described in Section 4.2).

We acknowledge that the feature weights can be difficult to interpret due to covariance among features. Indeed there are some unintuitive features, often negative indicators, in Table 4. We also see that the approach can produce unexpected results. For the *spicy* task, we see expected features like fresh chiles in the positive indicator list, but we also see sausage and shrimp. Sausage and shrimp do appear in many spicy recipes. Consequently, if one recipe has shrimp and another does not, it is reasonable to guess that the recipe with shrimp is more spicy. However, this is not an ideal solution because shrimp do not directly contribute to the spiciness of a dish.

Despite these issues, examining the feature weights provides a substantial amount of intuition about how predictions are made, and most of the important features are quite intuitive. For example, for ranking recipes according to the *picnic* attribute, the word "sandwiches" in the title is a positive indicator, while the presence of dairy is a negative indicator. For *creamy (soup)*, the presence of fresh cheese and cauliflower are positive indicators, while caramelization in the preparation steps and a large amount of water are negative indicators. For *refreshing (salad)*, cucumber and mint are positive indicators, while "warm" in the title is a negative indicator.

Table 5 displays held-out recipes with the most positive and negative scores. For example, for *refreshing (salad)*, our method predicts that "Easy Cucumber Salad" and "Fresh Fruit Salad with Honey, Mint and Lime Syrup" are refreshing while "San Antonio Taco Salad" and "Creamy Potato Salad with Grilled Scallions" are not. Similarly, "Soft Molasses Cookies" are chewy while "Easy Cut-Out Cookies" are not.

6 Applications

The proposed method can be used to improve the experience of finding and evaluating online recipes in several ways.

Recipe Tags: Predicted attributes can be displayed as "tags" on recipe pages. These tags can help users to decide whether to make a recipe.

Recipe Search: Predicted attributes can be matched with attributes identified in search queries to improve search relevance. For example, this would allow a retrieval system to find more relevant recipes for queries like "moist cake" and "refreshing salad." Importantly, our approach allows boost-

winter		refreshing (salad)		picnic	
title : squash	+1.122	title : cucumber	+0.839	title : sandwiches	+0.935
title : soup	+1.056	title : mint	+0.556	title : salad	+0.748
title : chowder	+0.862	ingr : fruits	+0.461	ingr : mozzarella cheese	+0.585
title : hot	+0.850	ingr : lemon peel	+0.460	amt : pork (+σ)	+0.559
ingr : seafood seasoning	+0.784	title : orange	+0.396	title : bars	+0.554
title : stew	+0.747	ingr : dill leaf	+0.300	prep : open	+0.510
title : chili	+0.653	ingr : flavorings (+2σ)	+0.283	title : corn	+0.498
title : gingerbread	+0.646	ingr : world herbs	+0.279	title : blueberry	+0.478
ingr : clove	+0.591	ingr : coriander	+0.275	ingr : instant vanilla pudding	+0.446
title : cinnamon	+0.563	title : tomato	+0.271	prep : chill	+0.444
ingr : shellfish	-0.733	ingr : bacon	-0.545	ingr : cream	-0.600
ingr : mayonnaise	-0.600	ingr : wheat flours	-0.522	prep : skillet	-0.409
prep : fry	-0.598	title : roasted	-0.497	ingr : milk	-0.375
ingr : chocolate	-0.579	prep : steam	-0.474	prep : paper towel	-0.319
prep : colander	-0.467	ingr : snap beans	-0.425	iamt : wheat flours (+σ)	-0.318
ingr : common tropical fruit	-0.463	ingr : parmesan cheese	-0.424	ingr : alcohol	-0.255
prep : soak	-0.390	prep : warm	-0.412	ingr : milk & cream	-0.222
ingr : extract	-0.384	title : warm	-0.373	ingr : asian condiments	-0.218
amt : poultry (+σ)	-0.353	title : potato	-0.360	prep : pan	-0.217
ingr : pasta filata	-0.343	ingr : pinto beans	-0.339	ingr : cheddar	-0.188

chewy (cookies)		creamy (soup)		spicy	
title : chewy	+1.455	ingr : nut/seed pastes	+0.829	title : spicy	+1.291
ingr : baking mix	+0.682	title : creamy	+0.694	title : sausage	+0.676
amt : nut/seed pastes (+σ)	+0.393	ingr : fresh cheeses	+0.645	ingr : hispanic condiments	+0.535
amt : fats (-2σ)	+0.378	title : split	+0.450	ingr : fresh chiles	+0.523
ingr : leavening agents	+0.344	title : cauliflower	+0.448	ingr : hot pepper sauce	+0.481
ingr : syrups	+0.335	ingr : wheat flours	+0.438	ingr : shrimp	+0.408
amt : milk/cream (+2σ)	+0.317	title : tomato	+0.370	ingr : pork sausages	+0.398
ingr : dark brown sugar	+0.284	title : squash	+0.355	title : carrot	+0.395
prep : bake	+0.271	ingr : soup	+0.336	title : green	+0.372
amt : white sugar (+σ)	+0.252	ingr : dairy	+0.298	ingr : ground cloves	+0.344
ingr : fruit vegetables	-0.507	title : barley	-0.384	amt : soup (+σ)	-0.464
amt : fats & oils (+σ)	-0.366	prep : caramelize	-0.363	prep : beat	-0.433
prep : cut	-0.342	title : vegetable	-0.352	amt : grain products (+2σ)	-0.421
amt : leavening agents (+2σ)	-0.337	ingr : stuffed pasta	-0.314	ingr : vinegar	-0.362
prep : knead	-0.319	ingr : thyme	-0.299	ingr : citrus fruit	-0.335
amt : wheat flours (+σ)	-0.289	title : onion	-0.272	prep : trim	-0.331
amt : sugars (-σ)	-0.275	ingr : cabbages	-0.255	ingr : vinegars	-0.326
prep : knife	-0.250	ingr : baked goods	-0.250	ingr : chocolate	-0.321
prep : crumble	-0.238	amt : water (+σ)	-0.242	ingr : dill leaf	-0.300
ingr : preserves/fruit butters	-0.214	ingr : sausages	-0.236	ingr : mushrooms	-0.296

Table 4: Features with the maximum and minimum weights for a sampling of the tasks. Training is conducted using all features. The abbreviations *ingr*, *amt*, and *prep* stand for ingredient, ingredient amount, and preparation step, respectively. Amount features with $(\pm c\sigma)$ denote that the amount is more than c standard deviations above or below the mean.

winter	
most positive:	
Quinoa with Moroccan Winter Squash and Carrot Stew	
Braised Provencal Chicken with Butternut Squash . . .	
Spicy Sausage Soup with Cilantro	
Spicy Peanut Soup with Chicken	
Italian Sausage and Tomato Soup	
most negative:	
Shrimp & Peppers Stir Fry	
Warm Jasmine Rice Salad with Shrimp and Thai Herbs	
Pan-Fried Cod with Slaw	
Garlic Bread Topped With Crab Meat and Spinach	
Singapore Chilli Prawns (Shrimp)	

refreshing (salad)	
most positive:	
Easy Cucumber Salad	
Fresh Fruit Salad with Honey, Mint and Lime Syrup	
German Cucumber Salad with Sour Cream	
Mango Pineapple Salad with Mint	
Cucumber Salad	
most negative:	
Warm Nut Encrusted Goat Cheese Salad with Bacon Lardons	
San Antonio Taco Salad	
Creamy Potato Salad with Grilled Scallions	
Farmers' Market Salad with Spiced Goat Cheese Rounds	
Sweet Corn and Basmati Rice Salad	

picnic	
most positive:	
Macaroni Salad with Peas and Ham	
Roast Beef Sandwiches with Lemon-Basil Mayonnaise	
Kittencal's Tuna Salad Sandwiches	
My Family's Tuna-Pasta Salad	
Radish Sandwiches	
most negative:	
Potato Soup V	
Smoky Four Cheese Macaroni Bake	
Baked Spaghetti	
The Best Butterscotch Banana Bread	
Chicken with Vin Jaune and Morels	

chewy (cookies)	
most positive:	
Soft and Chewy Peanut Butter Cookies	
Chewy Chocolate Cookies I	
Chewy Apple Oatmeal Cookies	
Chocolate Chewy Cookies	
Soft Molasses Cookies	
most negative:	
Pilgrim Hat Cookies	
Easy Cut-Out Cookies	
Walnut Butter Cookies	
Pill Bottle Cookies	
Chocolate Heart Cookies	

creamy (soup)	
most positive:	
Cream of Tomato Soup	
Butternut Squash Soup	
Cream of Potato Soup III	
Broccoli and Cheese Soup with Croutons	
Creamy Potato Leek Soup	
most negative:	
Vegetable and Ground Beef Soup	
Vegetable-Sausage Soup	
Ham Bone Vegetable Soup 1967	
Wild Mushroom and Barley Soup	
Beef Barley Soup	

spicy	
most positive:	
Spicy Shrimp and Grits	
Grant's Famous Midnight Grill BBQ Sauce	
Spicy Black Beans with Bell Peppers and Rice	
Spicy Filet Mignon	
Black Beans and Tomatoes - Hot and Spicy	
most negative:	
Spinach-Artichoke Ravioli-Lasagna	
Perfect Chocolate Cake	
Lemon Chiffon Pie with Gingersnap Crust	
Goat Cheese and Onion Tarts	
Whole White Wheat and Honey Chocolate Chip Cookies	

Table 5: Held-out recipes with the largest and smallest scores for different attributes. Training is conducted using all features.

ing the relevance score of a recipe even if the title does not contain the attribute and few or no reviews are available.

Recipe Recommendation: The predicted attributes can be used as inputs to recommendation systems. There is substantial ambiguity about what should be returned for a query like "salad." However, if we know that a user prefers "refreshing salads", then we know that "Potato Salad with Bacon" is probably not what was intended.

Recipe Modification: Often users would like to make a recipe but do not have all of the necessary ingredients. Our system can be used to predict how the omission of an ingredient is likely to change the recipe. For example, if a user has less oil than is suggested for a cake, we could compute the difference in the moistness attribute score after making the modification, and alert the user if the difference is large.

7 Discussion and Future Work

The primary advantage of using reviews as a source of supervision is that the data is essentially free. Creating manually annotated data sets for these tasks would be extraordinarily time consuming and expensive, as naively it would require preparing and testing thousands of recipes. Automatically generating attribute probabilities from user generated reviews allows the approach to scale to a large number of attributes at a significantly lower cost. However, the disadvantage of using reviews as a source of supervision is that the resulting data is noisy. There may be errors in identifying attributes in reviews, or the reviews may not mention the attribute even if it applies. In particular, we have noticed that in some cases an attribute applies to a recipe, but a user is unlikely to mention it in their review either because it is obvious or because it is not something that would occur to them. For example, some of the most negative *moist (cake)* predictions are for ice cream cakes. Ice cream cakes are moist, but reviewers are unlikely to mention this. However, this suggests that users may not expect or want ice cream cake recipes to be tagged as moist.

Some of our tasks include a dish filter. The motivation for this is that if an attribute is particularly associated with a dish (e.g. *chewy* is associated with *cookies*), then without a dish filter the model may simply learn to differentiate the associated dish from all other dishes. Additionally, an attribute might take on a particular meaning for a particular dish. A soup is likely to be bland for different reasons than a salad. In future work, we plan to introduce latent variables, or explore deep learning methods, so that the models have a non-linear decision boundary and can learn recipe groupings that are best for predicting a particular attribute without manual dish filtering.

We also plan to consider improved strategies for discovering and identifying recipe attributes. Ideally we would like to remove the manual steps described in Section 3. Instead, we are interested in methods that would automatically induce a taxonomy of attributes and taggers from the reviews, accounting for negation and synonyms. This would allow the method to more easily scale to large numbers of attributes.

Finally, the features of the preparation steps we use are not very effective (see Table 3). In future work, we plan to explore the use of a structured representation of the preparation steps. For example, instead of a feature for the word *mixed*, we would like to encode the ingredients that are mixed.

References

[Andrew and Gao, 2007] Galen Andrew and Jianfeng Gao. Scalable training of l1-regularized log-linear models. 2007.

[Badra *et al.*, 2008] Fadi Badra, Rokia Bendaoud, Rim Bentebibel, Pierre-Antoine Champin, Julien Cojan, Amélie Cordier, Sylvie Desprès, Stéphanie Jean-Daubias, Jean Lieber, Thomas Meilender, Alain Mille, Emmanuel Nauer, Amedeo Napoli, and Yannick Toussaint. Taaable: Text mining, ontology engineering, and hierarchical classification for textual case-based cooking. In *ECCBR Workshops*, pages 219–228, 2008.

[Bridge and Healy, 2012] Derek Bridge and Paul Healy. The ghostwriter-2.0 case-based reasoning system for making content suggestions to the authors of product reviews. *Know.-Based Syst.*, 29:93–103, May 2012.

[Burges *et al.*, 2005] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM.

[Druck and Pang, 2012] Gregory Druck and Bo Pang. Spice it up? mining refinements to online instructions from user generated content. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, 2012.

[Forbes and Zhu, 2011] Peter Forbes and Mu Zhu. Content-boosted matrix factorization for recommender systems: experiments with recipe recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, pages 261–264, New York, NY, USA, 2011. ACM.

[Freyne *et al.*, 2011] Jill Freyne, Shlomo Berkovsky, and Gregory Smith. Recipe recommendation: accuracy and reasoning. In *Proceedings of the 19th international conference on User modeling, adaption, and personalization*, UMAP'11, pages 99–110, 2011.

[Herbrich *et al.*, 2000] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. *Large margin rank boundaries for ordinal regression*. MIT Press, Cambridge, MA, 2000.

[Järvelin and Kekäläinen, 2002] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002.

[Joachims, 2002] Thorsten Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002.

[Liu, 2009] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, March 2009.

[Teng *et al.*, 2011] ChunYuen Teng, Yu-Ru Lin, and Lada A. Adamic. Recipe recommendation using ingredient networks. *CoRR*, abs/1111.3919, 2011.

[Ueda *et al.*, 2011] Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima. User's food preference extraction for cooking recipe recommendation. In *SPIM*, pages 98–105, 2011.

[Wang *et al.*, 2008] Liping Wang, Qing Li, Na Li, Guozhu Dong, and Yu Yang. Substructure similarity measurement in chinese recipes. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 979–988, 2008.

[Zhang *et al.*, 2008] Qian Zhang, Rong Hu, Brian Mac Namee, and Sarah Jane Delany. Back to the future: Knowledge light case base cookery. In Martin Schaaf, editor, *ECCBR Workshops*, pages 239–248, 2008.