Cooking with Computers 2013
Workshop Proceedings

Amélie Cordier, LIRIS, University of Lyon, France
Emmanuel Nauer, LORIA, University of Lorraine, France
Michael Wiegand, Saarland University, Germany

Program Committee

- Kerstin Bach, Verdande Technology, Norway
- Olivier Corby, INRIA, France
- Amélie Cordier, LIRIS, France
- Sylvie Despres, University of Paris 13, France
- Antoine Durieux, Chef Jérôme, France
- Takuya Funatomi, Kyoto University, Japan
- Willem Robert van Hage, Vrije Universiteit Amsterdam, Netherlands
- Ichiro Ide, Nagoya University, Japan
- Marie Lefevre, LIRIS, University of Lyon, France
- Mirjam Minor, Institut für Informatik, Germany
- Emmanuel Nauer, LORIA, France
- Thomas Roth-Berghofer, University of West London, UK
- Tomohide Shiabata, Kyoto University, Japan
- Jan Top, VU Amsterdam, Netherlands
- Michael Wiegand, Saarland University, Germany

Content

- Flavor Pairing in Medieval European Cuisine: A Study in Cooking with Dirty Data.

  Kush Varshney, Lav Varshney, Jun Wang and Daniel Myers.

- Learning ingredient space with generative probabilistic models.

  Vladimir Nedovic.

- Do Good Recipes Need Butter? Predicting User Ratings of Online Recipes.

  Ning Yu, Desislava Zhekova, Can Liu and Sandra Kübler.

- Recipe Attribute Prediction using Review Text as Supervision.

  Gregory Druck.

- Cooking Coach Spoken/Multimodal Dialogue Systems.

  Romain Laroche, Laurent Roussarie, Piotr Baczyk and Jan Dziekan.

Flavor Pairing in Medieval European Cuisine: A Study in Cooking with Dirty Data

Kush R. Varshney,[1] **Lav R. Varshney,**[1] **Jun Wang,**[1] **and Daniel Myers**[2]
[1]IBM Thomas J. Watson Research Center
1101 Kitchawan Road, Yorktown Heights, NY, USA
{krvarshn,lrvarshn,wangjun}@us.ibm.com
[2]Medieval Cookery
dmyers@medievalcookery.com

Abstract

An important part of cooking with computers is using statistical methods to create new, flavorful ingredient combinations. The flavor pairing hypothesis states that culinary ingredients with common chemical flavor components combine well to produce pleasant dishes. It has been recently shown that this design principle is a basis for modern Western cuisine and is reversed for Asian cuisine.

Such data-driven analysis compares the chemistry of ingredients to ingredient sets found in recipes. However, analytics-based generation of novel flavor profiles can only be as good as the underlying chemical and recipe data. Incomplete, inaccurate, and irrelevant data may degrade flavor pairing inferences. Chemical data on flavor compounds is incomplete due to the nature of the experiments that must be conducted to obtain it. Recipe data may have issues due to text parsing errors, imprecision in textual descriptions of ingredients, and the fact that the same ingredient may be known by different names in different recipes. Moreover, the process of matching ingredients in chemical data and recipe data may be fraught with mistakes. Much of the 'dirtiness' of the data cannot be cleansed even with manual curation.

In this work, we collect a new data set of recipes from Medieval Europe before the Columbian Exchange and investigate the flavor pairing hypothesis historically. To investigate the role of data incompleteness and error as part of this hypothesis testing, we use two separate chemical compound data sets with different levels of cleanliness. Notably, the different data sets give conflicting conclusions about the flavor pairing hypothesis in Medieval Europe. As a contribution towards social science, we obtain inferences about the evolution of culinary arts when many new ingredients are suddenly made available.

"Computational creativity is a subfield of artificial intelligence research ... where we build and work with computational systems that create art[i]facts and ideas. These systems are usually, but not exclusively, applied in domains historically associated with creative people, such as mathematics and science, poetry and story telling, musical composition and performance, video game, architectural, industrial and graphic design, the visual, and even the culinary, arts" [Colton and Wiggins, 2012].

1 Introduction

The cooking of food and human evolution are intertwined. One could go so far as to argue that it is cooking that makes us human. We are naturally drawn to foods high in fat and sugar because of the nourishment we received from such foods during our evolution in resource-poor environments. However, we are also drawn to foods with complex layers of balanced flavors composed through the art and science of cooking. It is these foods that we find delectable, delightful, and delicious.

Human flavor perception is very complicated, involving a variety of external sensory stimuli and internal states [Shepherd, 2006; Lim and Johnson, 2012]. Not only does it involve the five classical senses, but also sensing through the gut, and the emotional, memory-related, motivational, and linguistic aspects of food. In addition to the basic tastes: sweet, sour, salty, bitter, and umami, the smell of foods is the key contributor to flavor perception, which is in turn a property of the chemical compounds contained in the ingredients [Burdock, 2004]. There are typically tens to hundreds of different flavor compounds per food ingredient [Ahn *et al.*, 2011].

Other contributors to flavor perception among the classical senses are the temperature, texture, astringency, and creaminess of the food; the color and shape of food; and the sound that the food makes. The digestive system detects the autonomic and metabolic properties of the food. Moreover, there are emotion, motivation, and craving circuits in the brain that influence flavor perception, which are in turn related to language, feeding, conscious flavor perception, and memory circuits. Furthermore, effects beyond the food itself, including social and contextual ones, influence flavor perception [King *et al.*, 2013].

The key quality of foods and the one we focus on in this paper is the set of flavor compounds that they contain, which is a union of the flavor compounds of the constituent ingredients. Recent work has shown that olfactory pleasantness can be predicted based on the structure of flavor compounds [Khan *et al.*, 2007; Lapid *et al.*, 2008; Haddad *et al.*, 2010].

We note that cooking ingredients together can influence flavor perception as proteins, fats and starches bind certain flavor compounds and certain compounds may evaporate away or be chemically changed, but this is a second-order effect which we do not study in this work [Guichard, 2002].

One of the main guiding principles in putting together a recipe or a dish is flavor pairing. It is believed that ingredients that share many flavor compounds go well together. This flavor pairing hypothesis arose when the chef Heston Blumenthal found caviar and white chocolate to go well together, and investigated the basic chemical reason for why this is a good pairing [Blumenthal, 2008]. The flavor pairing hypothesis has been scientifically studied for several modern cuisines and found to hold strongly for Western cuisine, but to be almost opposite in East Asian cuisine [Ahn *et al.*, 2011].

In interviews that we conducted with professional chef instructors from the Institute of Culinary Education, we found that cutting-edge chefs today do truly think in terms of pairs or triplets of ingredients when coming up with new recipes. While they are not explicit about the chemistry of flavor compounds, they do draw on their mental databases of flavors and combinations to pair ingredients. They also draw upon other types of similarities between ingredients, such as being grown in the same season of the year or in the same region of the world.

One way in which we can use computers in cooking is as an aid in coming up with new ingredient pairings that may lie outside of the chef's mental repository. Having a machine system generate completely new flavor combinations that have never existed before is a culinary application of computational creativity [Bhattacharjya *et al.*, 2012; Morris *et al.*, 2012; Veeramachaneni *et al.*, 2012; Varshney *et al.*, 2013]. Such a culinary computational creativity system or synthetic gastronomist depends critically on its data repository (just like human culinary creators).

The quality of data (or lack thereof) is a prevalent issue in almost all fields of analytics [Kim *et al.*, 2003; De Veaux and Hand, 2005; Wang *et al.*, 2012]. Here we list several aspects of this issue as they specifically pertain to the culinary domain approached with computers. One issue is resolving the names of ingredients that refer to the same entity, e.g. 'emmental,' 'emmenthal' and 'emmenthaler'; 'bow tie pasta,' 'bowtie pasta' and 'farfalle'; 'cilantro,' 'coriander' and 'dhania'; 'New Mexico red chile' and 'red New Mexico chile'; and 'achiote,' 'annatto' and 'annatto seed.' In parsing and analyzing semi-structured recipe text, there can be issues in determining what the actual ingredient is, e.g. in 'my low-carb catsup,' and other aspects of data dirtiness arising from imperfect text analytics.

Moreover, the analytical chemistry experiments conducted to determine the flavor compounds present in a food ingredient are far from perfect and repositories far from complete. Thermal desorption–gas chromatography–mass spectrometry is the typical technique used to identify and sometimes quantify volatile flavor compounds in foods and beverages, but also in many forensics, monitoring, and other applications. Experiments have not been conducted and verified for every possible food ingredient, which presents a missing data limitation. Also, many compounds occur in trace amounts in foods; this contributes to false alarm and missed detection errors in chemical analysis. Moreover, matching food ingredients in recipes to food ingredients that have been chemically analyzed is another process that can introduce error and incompleteness. Thus overall, recipe, ingredient, and flavor compound data is plagued by various kinds of data dirtiness.

In order to cook with computers, specifically by having the computer use the flavor compound pairing hypothesis to create new sets of ingredients in recipes, we must understand the effect of data dirtiness. Towards that end, in this paper we obtain two independent data sets of flavor compound data having different levels of completeness and accuracy, and compile a completely new corpus of recipes from the Late Middle Ages that has not been analyzed before. We test the flavor pairing hypothesis twice using this corpus: once with each flavor compound data set. With one data set we see a positive confirmation of the flavor pairing hypothesis that is off the charts, but see an opposite result using the other data set. These conflicting conclusions arising from differences in data quality are quite notable and must be scrutinized in the process of cooking with computers.

A second contribution of our work beyond understanding dirty data when cooking with computers is a statement about the evolution of cuisine [Kinouchi *et al.*, 2008]. The medieval period represents an age just before the set of possible ingredients for European cooking increased dramatically. In the age of discovery that followed, European exploration of the Americas initiated the Columbian Exchange [Crosby, 1972], the transfer of animals, plants, bacteria, viruses, and culture between the continents. This transfer represents the most significant global event in terms of agriculture and cooking because it injected many now-common ingredients such as tomatoes and potatoes into Eurasian and African cuisine. When compared to modern Western cuisine using the same flavor compound data set, we find that the level of flavor pairing is approximately the same as in medieval cuisine, but that the ingredients available do not lend themselves as much to chemical pairing.

The remainder of this paper is organized as follows. In Section 2, we discuss the flavor pairing hypothesis in greater depth and describe the statistical methodology for testing it and the data required for such evaluation. Then we discuss the medieval period in Europe and the Columbian Exchange in Section 3. Section 4 is devoted to empirical methodology and results: testing flavor pairing on medieval recipes using two different flavor compound data sets. The main result in this section is that data quality issues can yield conflicting inferences. We provide discussion and conclude in Section 5.

2 Flavor Pairing Hypothesis Testing

As mentioned in the introduction, flavor pairing is a key concept in culinary arts that can be examined scientifically by investigating the chemical flavor compounds that are components of food ingredients. In this section, we discuss the different aspects of testing and quantifying the extent to which ingredients with common flavor compounds go well together.

2.1 Chemical Components

A strong determinant of the flavor of foods is the aromatic compounds that reach the olfactory system, either through the nose or through retro-olfaction. Humans are adept at detecting even trace amounts of these compounds and they have a great effect on hedonic perception. Chemically, flavor compounds come from groups such as acetals, acids, alcohols, aldehydes, esters, furans, hydrocarbons, ketones, lactones, and phenols.

The processes by which one can determine the flavor components of a food are a branch of analytical chemistry. The typical experiment involves heating the food to release the flavor compounds into a vapor. This gas is then passed through a gas chromatograph, which separates different types of molecules of the gas based on the time it takes them to travel through a capillary. The separated gas molecules are then analyzed using a mass spectrometer, which ionizes the molecules and measures mass-to-charge ratios to determine the elemental composition.

In such a manner, the flavor compounds that are present in a food dish, or more typically food ingredient, are identified. However, when many different flavor compounds are present in small quantities, the experiments generally have some false alarms and some missed detections. In any case, the result of the analytical chemistry is a list of contained compounds for the food ingredient under consideration. In our work, we consider two flavor compound databases: the Volatile Compounds in Food 14.1 database (VCF) and Fenaroli's Handbook of Flavor Ingredients as processed and released in [Ahn *et al.*, 2011].

2.2 Recipe Collections

Recipes in cookbooks represent the culinary best practices of a culture. As such, when taking a data-driven approach to understanding pairing, they also represent sets of ingredients that are flavorful together. Modern recipes list ingredients, but also quantities and instructions for preparation. In the medieval period, recipes were primarily only the former. For the purposes of analyzing pairing, it is only the set of ingredients that is of importance. If the flavor pairing hypothesis holds, then sets of ingredients in actual recipes should have, on average, more shared flavor compounds than any random set of ingredients. It is such a test that is proposed in [Ahn *et al.*, 2011] and that we conduct in this work.

We note that rather than using a large-scale statistical methodology with the foundational assumption that recipes in cookbooks represent the distillation of what people like and dislike, another approach is experimental. In a very small-scale sensory testing experiment, twenty-one food pairings involving pear, tomato, potato, chocolate, beef, cauliflower, and anise were made into purées and tested using human flavor perception experiments. Combinations with strong flavor pairing according to the VCF database were not necessarily the best rated by undergraduate test subjects [Kort *et al.*, 2010].

In this paper, we compile a corpus of recipes from Medieval Europe and analyze the flavor pairing hypothesis within this new collection. There are some notable problems with drawing conclusions based on recipes in cookbooks from the time if one's goal is to understand daily life in medieval times. Most notably, the cookbooks are of wealthy landowners, and thus do not necessarily reflect the diet of the poor or middle class. Also, the connection between recipes and what was actually cooked is open to question. However, our goal is to understand the most flavorful foods that were being concocted in that time and place, and recipes are an excellent source for that purpose.

2.3 Statistical Methodology

As described in [Ahn *et al.*, 2011], the primary calculation to understand the flavor pairing hypothesis is to compute the average number of shared flavor compounds among the ingredients in a recipe R. Let R be a set of n_R different ingredients. Then the average number of shared compounds is:

$$N_s(R) = \frac{2}{n_R(n_R-1)} \sum_{i,j \in R, i \neq j} |C_i \cap C_j|, \qquad (1)$$

where C_i is the set of flavor compounds in ingredient i and C_j is the set of flavor compounds in ingredient j. The mean of $N_s(R)$ across the corpus of recipes, which we denote $\bar{N}_s$, represents the degree to which flavor pairing exists overall.

Then in order to understand whether $\bar{N}_s$ is indicative of ingredients with high compound sharing also often appearing together in recipes (and thus implicitly tasting good together), we must compare $\bar{N}_s$ for the recipe corpus under consideration to a null hypothesis, specifically the value of $\bar{N}_s$ for randomly generated sets of ingredients from the same overall universe of ingredients and probability distribution. Denoting the average sharing for the true corpus of recipes as $\bar{N}_s^{\text{real}}$ and for a randomly generate corpus as $\bar{N}_s^{\text{rand}}$, the difference

$$\Delta N_s = \bar{N}_s^{\text{real}} - \bar{N}_s^{\text{rand}} \qquad (2)$$

if positive indicates that flavor pairing is a strong influence in the real recipes under consideration, if close to zero indicates no relationship between flavor compounds and recipes, and if negative indicates that recipes tend to include ingredients that spcifically do not share flavor compounds.

Additionally, as discussed extensively in [Ahn *et al.*, 2011], it is possible to calculate how much an individual ingredient i contributes to ΔN_s as follows:

$$\chi_i = \frac{1}{N_c} \sum_{R \ni i} N_s(R) - \left(\frac{2 f_i}{N_c \bar{n}_R} \frac{\sum_{j \in c} f_j |C_i \cap C_j|}{\sum_{j \in c} f_j} \right), \qquad (3)$$

where N_c is the number of recipes in the corpus, f_i is the number of occurrences of ingredient i, and $\bar{n}_R$ is the average number of ingredients per recipe in the corpus.

3 Medieval Times and the Columbian Exchange

The medieval period in Europe, also known as the Middle Ages, is the time between the collapse of the Western Roman Empire and the beginning of the Renaissance. The exact dates are a bit hard to pin down, and strongly depend on what part of Europe is being considered. For example the fifteenth century is considered as the Renaissance in Italy, but is the Late Middle Ages in England.

Cereal grains (barley, oats, and rye for the poor and wheat for the wealthy) were the main staples and were prepared as bread, porridge, gruel, and pasta. The staples were supplemented by vegetables. Meat was more expensive and eaten less, with pork and chicken being more prevalent than beef. Fish was common, especially cod and herring, but also other saltwater and freshwater fish. Wild game was common only among the nobility.

A misconception about that time period is that spices were used to cover the taste of spoiled meat. This myth has its origins in Victorian-era England and has no basis in fact. Such a practice would have been unfeasible in terms of health (it would have killed the people), economics (it would have been too expensive), and logistics (it would have required vast amounts of meat to be kept hanging for days).

The medieval period was an age prior to the exploration of the Americas. Once the New World had been discovered, many new ingredients made their way to Eurasia and vice versa. This transfer of foods, along with the transfer of diseases and culture is known as the Columbian Exchange after Christopher Columbus [Crosby, 1972; Nunn and Qian, 2010]. Some key ingredients that were absent in the Old World before the Columbian Exchange include corn, potatoes, cassava, sweet potatoes, tomatoes, sunflower seeds, cacao beans, pineapples, peanuts, eggplants, tobacco, vanilla, and capsicum peppers (which are the ancestors of cayenne peppers, bell peppers, and jalapeño peppers). Crops such as tomatoes, cacao, and chili peppers are not themselves especially rich in calories, but complement existing foods by increasing vitamin intake and improving flavor.

Often New World foods have had an important effect on cuisine evolution: chili peppers led to spicy curries in India, paprika in Hungary, and spicy kimchee in Korea; tomatoes significantly altered the cuisine of Italy and other Mediterranean countries. Thus it is interesting to examine cuisine from before the exchange to understand culinary evolution [Kinouchi *et al.*, 2008].

Book	Country	Date
MS B.L. Royal 12.C.xii	England/ France	1340
Forme of Cury	England	1390
Ancient Cookery	England	1425
Liber cure cocorum	England	1430
Two Fifteenth-Century Cookery Books	England	1450
A Noble Boke off Cookry	England	1468
Thomas Awkbarow's Recipes [MS Harley 5401]	England	15th c.
Gentyll manly Cokere [MS Pepys 1047]	England	ca. 1500
A Proper newe Booke of Cokerye	England	1550
A Book of Cookrye	England	1591
The Good Housewife's Jewell	England	1596
Delights for Ladies	England	1609
A NEVV BOOKE of Cookerie	England	1615
Enseignements	France	1300
Le Viandier de Taillevent	France	1380
Le Menagier de Paris	France	1393
Du fait de cuisine	France	1420
Le Recueil de Riom	France	15th c.
Ouverture de Cuisine	France	1604
Ein Buch von guter spise	Germany	1345
Das Kochbuch des Meisters Eberhard	Germany	1450
Das Kuchbuch der Sabina Welserin	Germany	16th c.
Libro di cucina / Libro per cuoco	Italy	14th/15th c.
The Neapolitan recipe collection	Italy	15th c.
Due Libri di Cucina - Libro B	Italy	15th c.

Table 1: Medieval source texts.

4 Empirical Methodology and Results

In this section, we describe the steps we undertook to empirically study the flavor pairing hypothesis in Medieval Europe, from constructing the corpus of recipes all the way to conducting the analytics. These are the same steps that need to be performed when cooking with a computer that suggests new food pairings based on cultural artifacts and chemistry.

4.1 Medieval Recipe Corpus Creation

The first step in investigating the flavors of Medieval Europe was to compile a collection of recipes from that age. In particular, we found recipes in twenty-five different source texts from England, France, Germany, and Italy, from the years 1300 to 1615. These cookbooks are listed in Table 1. Next, concordances were generated from the source texts. From these word lists, ingredient terms were identified, and the remaining parts of speech were discarded.

The terms were then manually placed into one of 391 equivalence groupings based upon plurality (e.g. 'cheese' and 'cheeses'), synonyms (e.g. 'mallard' and 'duck'), spelling variations (e.g. 'chicken' and 'chekin'), and foreign loan words (e.g. 'eyren' and 'eggs'). These last two types of grouping were made necessary by the inclusion of source texts written in Middle English. To build the ingredients lists, each source text was split into individual recipes. The recipes were compared against the table of equivalence groupings, with words not in the table being discarded. Found words were replaced with a lemmatized equivalent for consistency, with duplicates within a recipe being removed. Several examples of medieval recipes are given in Table 2. Upon visual inspection, the sets of ingredients are quite different than what one experiences today. The recipe ingredient preparation procedure was done as conscientiously as possible, but is not without error.

In total, the medieval cookbooks contained 4,133 recipes. After the text processing and word discardal, 41 recipes in our corpus are rendered blank. The corpus contains 386 different ingredients ranging from acorn to zedoary. The distribution

(a)	(b)	(c)	(d)	(e)
bean	venison	eel	mallard	frumenty
broth	wine	fish	bread	porpoise
onion	sage	bone	vinegar	almond
saffron	parsley	date	blood	milk
	hyssop	cod	pepper	
	pepper	almond	ginger	
	clove	milk		
	cinnamon	sugar		
	blood	maces		
		flour		
		rice		
		saffron		
		sandalwood		
		ginger		

Table 2: Five examples from our corpus of 4133 medieval recipes: (a) drawen benes, (b) roo in sene, (c) eles in brasill, (d) sause neyger for maudelard roasted, and (e) furmente with purpeys.

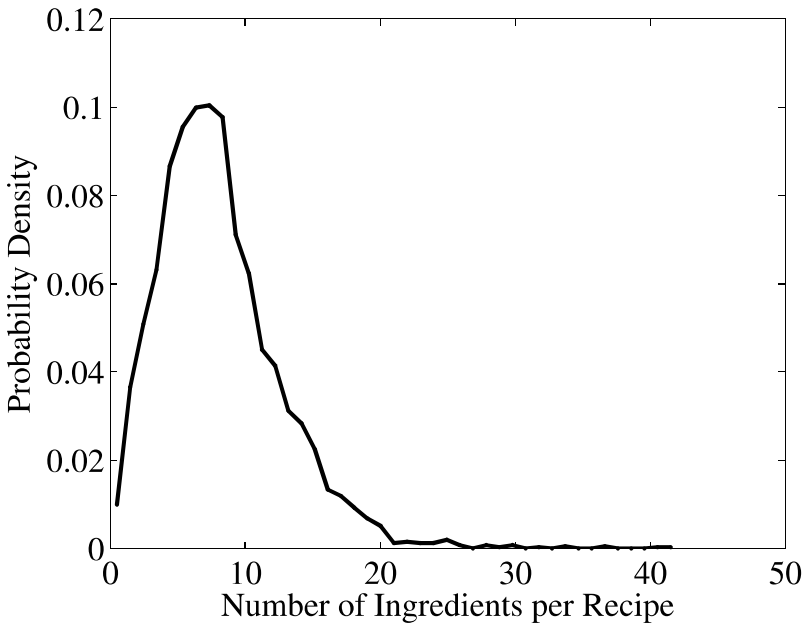

Figure 1: Probability density function of the number of ingredients per recipe in our medieval corpus.

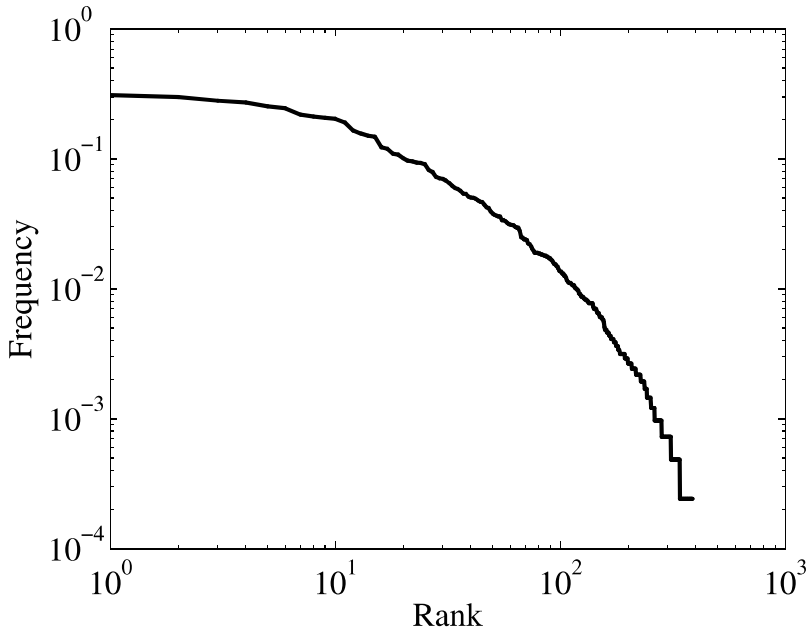

Figure 2: Rank frequency plot of ingredients in our medieval corpus.

of the number of ingredients per recipe is shown in Fig. 1. The mean is 7.74 ingredients per recipe, the maximum is 42, and the standard deviation is 4.60. A rank frequency plot of the ingredients in the recipe corpus is given in Fig. 2.

Medieval recipes were chosen for several reasons. One reason is that they have much historical interest. In fact, one of the chefs that we interviewed, Michael Laiskonis, specifically mentioned reading historical cookbooks as inspiration for new dishes. Another reason is that the medieval primary sources are in the public domain. In contrast, the recipe sources in [Ahn *et al.*, 2011], e.g. allrecipes.com, are proprietary and data extracted from them cannot be released publically.

4.2 Flavor Compound Data

As we have discussed, we are interested in examining the effect of data sets with different properties, and thus we conduct empirical studies with two different flavor compound databases: VCF and Fenaroli. The previous work on the flavor pairing hypothesis used only Fenaroli [Ahn *et al.*, 2011]. The first iteration of VCF, the Lists of Volatiles, was compiled by Weurman in 1963 at the Central Institute for Food Research, which is part of the Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (TNO). It is continually updated and enhanced by analytical chemists at TNO. We use version 14.1 which was released in January 2013. We scraped and parsed the flavor compound data from the online repository http://www.vcf-online.nl.

VCF contains 522 food products and 102 food product categories, which we take together as 624 ingredients. It also contains 7,647 unique flavor compounds. Examples of flavor compound listings per ingredient are given in Fig. 3(a) and Fig. 4(a) for saffron and almond respectively. The distribution of the number of flavor compounds per ingredient is shown in Fig. 5. The minimum number is 1 (lobster), the maximum is 2,733 (wine), the median is 83, the mean is 175.95, and the standard deviation is 285.93. A rank frequency plot of compounds in VCF is given in Fig. 6.

The other flavor compound source is the fifth edition of Fenaroli's Handbook of Flavor Ingredients [Burdock, 2004] as processed by [Ahn *et al.*, 2011]. The first edition of this work was published in 1971 and there also now exists a sixth edition. This data set has 1,530 ingredients and 1,107 flavor compounds. The distribution of the number of compounds per ingredient is shown in Fig. 7. The maximum number of flavor compounds per ingredient is 239 (black tea), the median is 2, the mean is 24.04, and the standard deviation is 43.07. The rank frequency plot is given in Fig. 8. The compound lists for saffron and almond from Fenaroli are given in Fig. 3(b) and Fig. 4(b).

Fenaroli has a greater number of ingredients than VCF but a much smaller number of flavor compounds. The average number of compounds per ingredient detected by Fenaroli is also much less than VCF. The mean to standard deviation ra-

2-phenylethanol (=phenethyl alcohol)
safranal (=2,6,6-trimethyl-1,3-cyclohexadienecarbaldehyde)
3,5,5-trimethyl-2-cyclohexen-1-one (=isophorone)
hexadecanoic acid (=palmitic acid)
2,6,6-trimethyl-2-cyclohexene-1,4-dione
(Z,Z)-9,12-octadecadienoic acid (=linoleic acid)
(Z,Z,Z)-9,12,15-octadecatrienoic acid (=linolenic acid)
naphthalene
2,4,6-trimethylbenzaldehyde (=mesitylaldehyde)
2,6,6-trimethyl-1,4-cyclohexadienecarbaldehyde
6,6-dimethyl-2-methylene-3-cyclohexenecarbaldehyde
4-hydroxy-2,6,6-trimethyl-1-cyclohexenecarbaldehyde (=4-hydroxysafranal)
3,5,5-trimethyl-3-cyclohexen-1-one
3,3,4,5-tetramethylcyclohexanone
3,5,5-trimethyl-4-methylene-2-cyclohexen-1-one
4-hydroxy-3,5,5-trimethyl-2-cyclohexen-1-one
2,3-epoxy-4-(hydroxymethylene)-3,5,5-trimethylcyclohexanone
5,5-dimethyl-2-cyclohexene-1,4-dione
2,2,6-trimethylcyclohexane-1,4-dione (=3,5,5-trimethyl-cyclohexane-1,4-dione)
2-hydroxy-3,5,5-trimethyl-2-cyclohexene-1,4-dione
2-hydroxy-4,4,6-trimethyl-2,5-cyclohexadien-1-one
2,6,6-trimethyl-3-oxo-1,4-cyclohexadienecarbaldehyde
4-hydroxy-2,6,6-trimethyl-3-oxo-1,4-cyclohexadienecarbaldehyde
4-hydroxy-2,6,6-trimethyl-3-oxo-1-cyclohexenecarbaldehyde
3-hydroxy-2,6,6-trimethyl-4-oxo-2-cyclohexenecarbaldehyde
4-(2,2,6-trimethyl-1-cyclohexyl)-3-buten-2-one
4-(2,6,6-trimethyl-1-cyclohexen-1-yl)-3-buten-2-one (=β-ionone)
verbenone (=2-pinen-4-one)
octadecanoic acid (=stearic acid)
(Z)-9-octadecenoic acid (=oleic acid)
2(5H)-furanone (=crotonolactone, 2-buten-4-olide, 4-hydroxy-2-butenoic acid lactone)

(a)

phenethyl alcohol
2,6,6-trimethylcyclohexa-1,3-dienyl methanal
isophorone
palmitic acid
2,6,6-trimethylcyclohex-2-ene-1,4-dione
9,12-octadecadienoic acid (48%) plus 9,12,15-octadeca- trienoinc acid (52%)

(b)

Figure 3: Flavor compounds in saffron (Crocus sativus L.) from (a) VCF data set and (b) Fenaroli data set. The compounds in bold appear in both lists.

tio for both sets is similar and the shape of the distributions and rank frequency plots is also similar. The real key difference is in the coverage of the data sets reflected in the axis labels of the plots. Looking at the two example ingredients, saffron and roasted almond, we see that all compounds listed for saffron in Fenaroli also appear in VCF,[1] but there are additional compounds in VCF. Similarly, most Fenaroli compounds for roasted almond appear in VCF whereas VCF has a greater number that do not appear in Fenaroli. We examine the effect of such a differences in data on quantification of flavor pairing in Section 4.4.

4.3 Ingredient Matching

The final piece of data preparation is matching the names of ingredients from the medieval recipes and the two chemical compound data sets. For Fenaroli, we used the ingredient names of [Ahn *et al.*, 2011] and did a simple string match to the ingredient names in our medieval corpus. We were able to match 157 ingredients and were unable to match 229 ingredients. We note that Ahn et al. associate the compounds

[1]The names of compounds may not match exactly, but are matched chemically using the Chemical Abstracts Service registry. The orders of the matching molecules correspond in the tables.

α-ionone
2-acetylpyrrole (=methyl 2-pyrrolyl ketone)
phenol (=hydroxybenzene)
furfuryl alcohol (=(2-furyl)-methanol, 2-furanmethanol)
methyl 2-furancarboxylate
furfuryl acetate (=2-furanmethanol acetate)
6,7-dihydro-5-methyl-5H-cyclopentapyrazine
3-methyl-1,2-cyclopentanedione (=cyclotene)
trimethylpyrazine
hexane
benzaldehyde
4-hydroxy-4-methyl-2-pentanone (=diacetone alcohol)
(E)-β-ionone
2-pyrrolecarbaldehyde (=2-formylpyrrole)
(2-furyl)pyrazine
2,5-dimethylpyrazine
2,6-dimethylpyrazine
2-(2-furyl)-3-methylpyrazine
furfural (=2-formylfuran, 2-furancarbaldehyde, 2-furaldehyde)
5-(hydroxymethyl)furfural
4-hydroxy-2-(hydroxymethyl)-5-methyl-3(2H)-furanone
2-acetylfuran (=2-furyl methyl ketone, 1-(2-furyl)ethanone)

(a)

a-ionone
methyl-2-pyrrolyl ketone
phenol
furfuryl alcohol
methyl furoate
furfuryl acetate
5h-5-methyl-6,7-dihydrocyclopenta(b)pyrazine
methylcyclopentenolone
2,3,5-trimethylpyrazine
acetylpyrazine
1-tyrosine
b-ionone
l-histidine

(b)

Figure 4: Flavor compounds in almond (roasted) (Prunus amygdalus) from (a) VCF data set and (b) Fenaroli data set. The compounds in bold appear in both lists.

in essential oils and extracts to the original ingredient and include the flavor compounds of more general ingredients into more specific ingredients. For VCF, we did manual processing of the ingredients and were able to match 191 ingredients. In VCF, ingredients are sometimes part of larger ingredient categories; following a similar philosophy as Ahn et al., we matched to categories when possible. One major factor in the higher VCF match rate is that the medieval corpus contains forty-two different fish, e.g. anchovy, bass, dace, hake, and whiting. We matched all of these fish to the VCF product category fish. Although the overall database of Fenaroli has more ingredients than VCF, the difference is no longer significant after matching to the medieval corpus.

4.4 Flavor Pairing Analysis

With all data collected, prepared, and matched, we can perform the statistical analysis described in Section 2.3. We first calculate the average number of shared compounds among the over four thousand recipes in our medieval corpus. The distribution of $N_s(R)$ using the Fenaroli data is shown in Fig. 9. The distribution using the VCF data is shown in Fig. 10. The average across the corpus is calculated as $\bar{N}_s^{\text{real}} = 11.26$ for Fenaroli and $\bar{N}_s^{\text{real}} = 51.42$ for VCF.

The values are quite different due to VCF containing so many more flavor compounds. We can examine how corre-

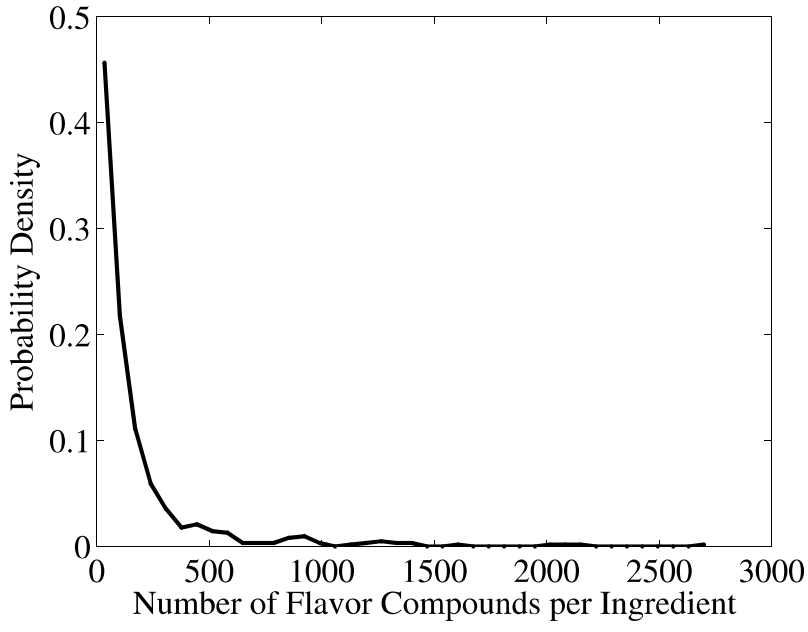

Figure 5: Probability density function of the number of compounds per ingredient in VCF.

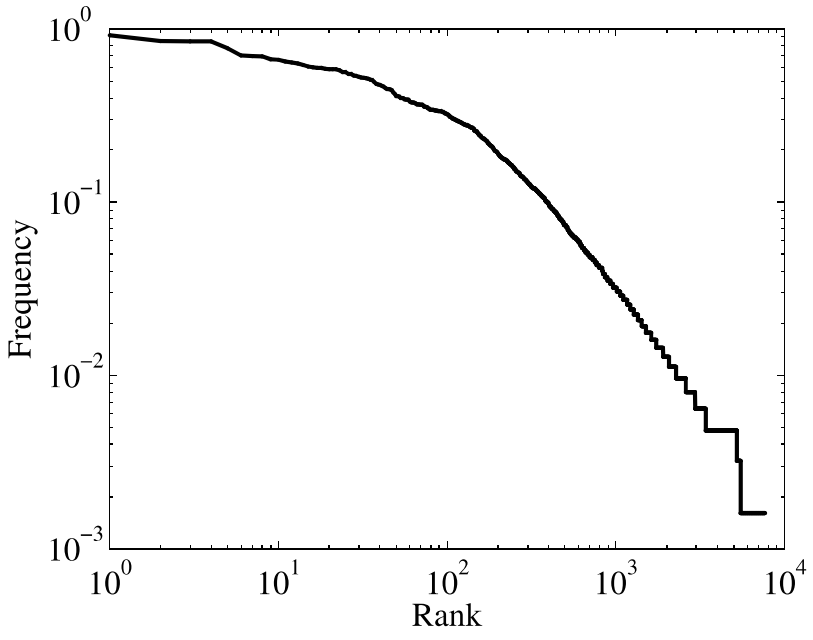

Figure 6: Rank frequency plot of compounds in VCF.

lated the $N_s(R)$ values are when using the two different flavor compound data sets. A scatter plot is shown in Fig. 11. The correlation coefficient between the two is 0.542. Since most of the $N_s(R)$ values are small, we can also examine the correlation after a logarithmic transformation. The correlation coefficient between the $\log(N_s(R)+1)$ values using the two data sets is 0.688. The shared compound calculation using the two different flavor compound data sets yields similar results, but not a full correlation.

Now to understand the meaning of the $\bar{N}_s^{\text{real}}$ values, we must also calculate $\bar{N}_s^{\text{rand}}$ for the two compound sets. Using the exact same instantiation of random medieval ingredient sets, we find $\bar{N}_s^{\text{rand}} = 4.54$ for Fenaroli and $\bar{N}_s^{\text{rand}} = 54.17$ for VCF, yielding $\Delta N_s = 6.72$ for Fenaroli and $\Delta N_s = -2.75$ for VCF.

These delta values lead to opposite conclusions. On one hand, using the Fenaroli data, we see a very strong positive indicator of the flavor pairing hypothesis in Medieval Europe.

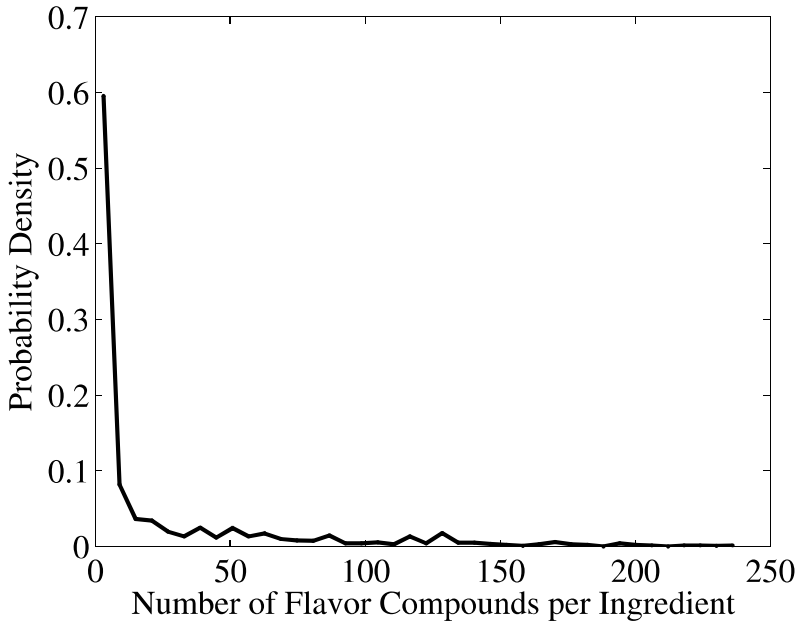

Figure 7: Probability density function of the number of compounds per ingredient in Fenaroli.

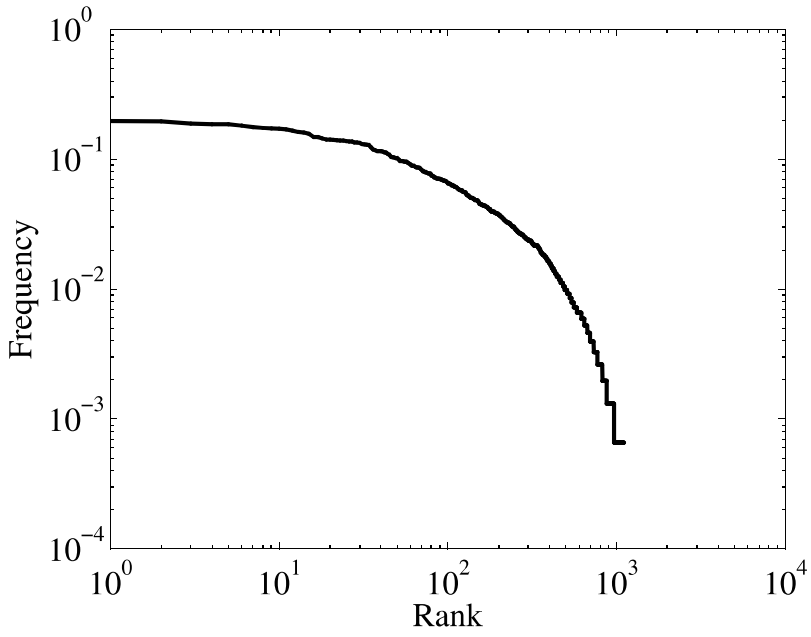

Figure 8: Rank frequency plot of compounds in Fenaroli.

A value of 6.72 is much larger than for any modern cuisine reported in [Ahn *et al.*, 2011]. (The $\bar{N}_s^{\text{real}}$ value for Medieval Europe is actually quite similar to modern North America, but the $\bar{N}_s^{\text{rand}}$ value for Medieval Europe is smaller.) On the other hand, using the VCF data, we obtain a negative ΔN_s, which means that ingredients that don't share many flavor compounds are used together.

We can also calculate and examine the individual ingredient contributions. Table 3 list the top and bottom fifteen contributors using the two data sets. The bottom fifteen contributors are fairly stable with respect to the two chemical databases, but the top fifteen contributors are different. The VCF list is dominated by fish, whereas the Fenaroli list does have many fish, but other things as well. Most of the fish at the top of the VCF list were not matched using Fenaroli data. This difference may be the main contributor for the conflicting results.

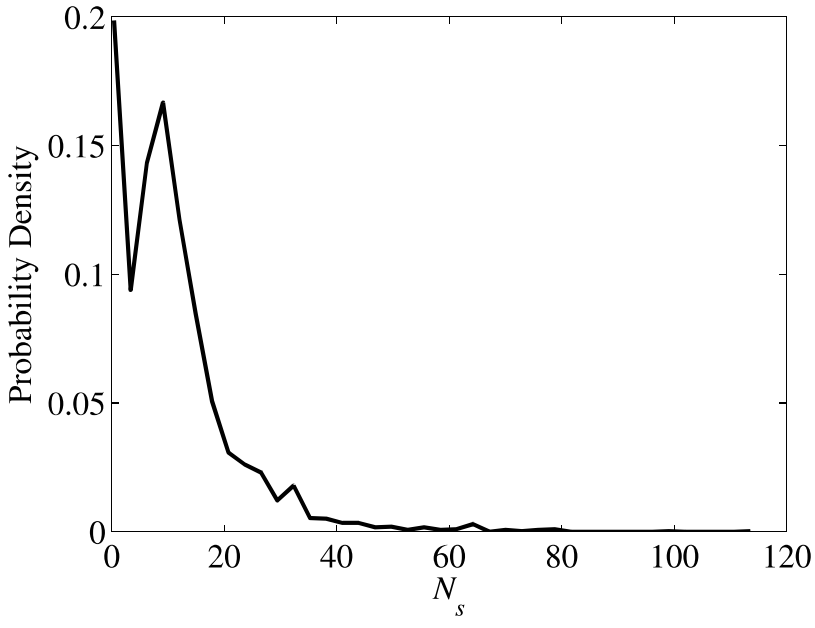

Figure 9: Probability density function of the number of average shared flavor compounds per recipe from the Fenaroli data set. The mean $\bar{N}_s = 11.26$.

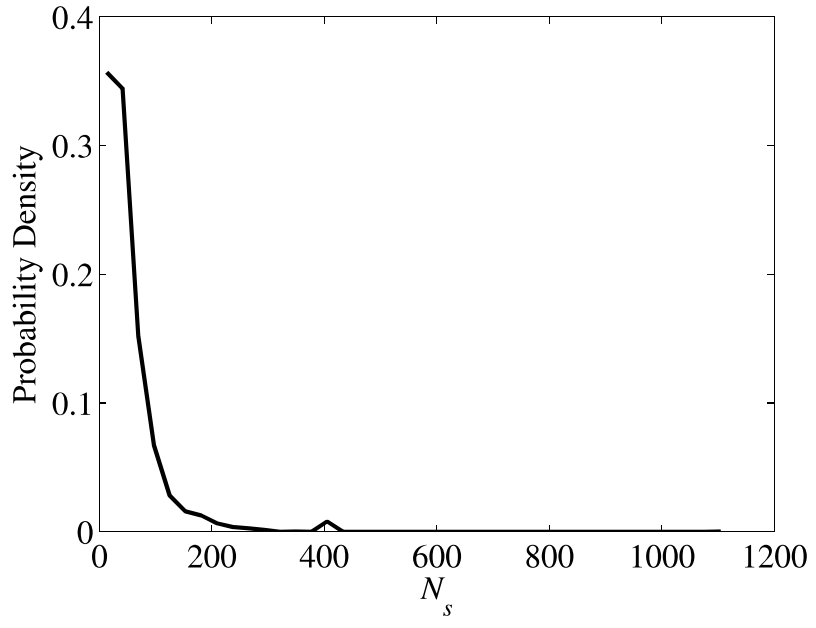

Figure 10: Probability density function of the number of average shared flavor compounds per recipe from the VCF data set. The mean $\bar{N}_s = 51.42$.

	Top Fen	Top VCF	Bottom Fen	Bottom VCF
1	whale	halibut	filbert	valerian
2	blackberry	dace	lentil	buttermilk
3	bacon	thorneback	octopus	horseradish
4	haddock	sole	valerian	eggplant
5	tuber	hake	horseradish	oregano
6	beer	turbot	caviar	chicory
7	salmon	mullet	oregano	cuttlefish
8	cider	carp	chickpea	caviar
9	beef	dogfish	cuttlefish	clam
10	strawberry	ray	vervain	lentil
11	cod	shad	nettle	barley
12	herring	trout	buttermilk	turkey
13	cheese	citron	clam	minnow
14	grape	gurnard	pennyroyal	prawn
15	bean	bream	rue	scallop

Table 3: Top and bottom fifteen contributing ingredients to medieval cuisine.

5 Conclusion

In this work, we have examined food ingredients that appear in medieval recipes, focusing on how many chemical flavor compounds the ingredients share. Our contribution is studying the reasons and effects of dirty data, in particular finding that conclusions can be reversed by differences in data quality. Specifically, we have tested the hypothesis that food ingredients that share many flavor compounds go together in dishes.

Using a sparser and more incomplete chemical database and matching procedure, we find the hypothesis to be true in Medieval Europe. Moreover, in comparing with analysis of modern regional cuisines using the same exact chemical database, we find the pairing to be stronger in the medieval period than in modern times. The main difference is not in the level of pairing in the recipes, but in the lack of potential pairing in the available ingredients as expressed through random recipe ensembles. As is known historically, the number of ingredients available after the Columbian Exchange, including in the modern world, is much greater than before. The results we obtain bear this fact out. Even though medieval cooks had a more difficult job because there were fewer paired ingredients, they were able to achieve the same level of flavor compound pairing. After the exchange and the introduction of a boatload of new ingredients, Western cooks have maintained the pairing level, but increased variety. We can conjecture that there is some combination of pairing and variety or balance that chefs aim to achieve; by having more ingredients, they are able to more easily satisfy the pairing and turn their attention to variety and balance.

In future work, it would be interesting to see whether this inference, comparison, and conjecture holds when analyzing modern recipes using the more complete chemical database we have utilized to study medieval cuisine. We have seen here that the quality of the raw data and quality of the data preparation have a fundamental downstream effect on analysis. The more complete dataset has indicated the opposite: that ingredients with shared compounds are not over-represented in recipes. Understanding this result requires more detailed study.

Scientifically validated design principles are important for using computational techniques in generating flavorful, novel, and healthy culinary recipes. These design principles, however, are often derived from large-scale data analysis, and so there is a need for complete and accurate data sources. In this work on medieval cuisine, we found different results using different flavor compound databases. From the point of view of computational creativity for culinary recipes, if we want to generate foods that people from Medieval Europe might find flavorful, this leaves us in a bit of a dilemma. Should we be promoting flavor pairing or not.

Broadly speaking, computational creativity algorithms have two phases: first generating combinatorially many new ideas, and then evaluating the ideas on metrics of quality and

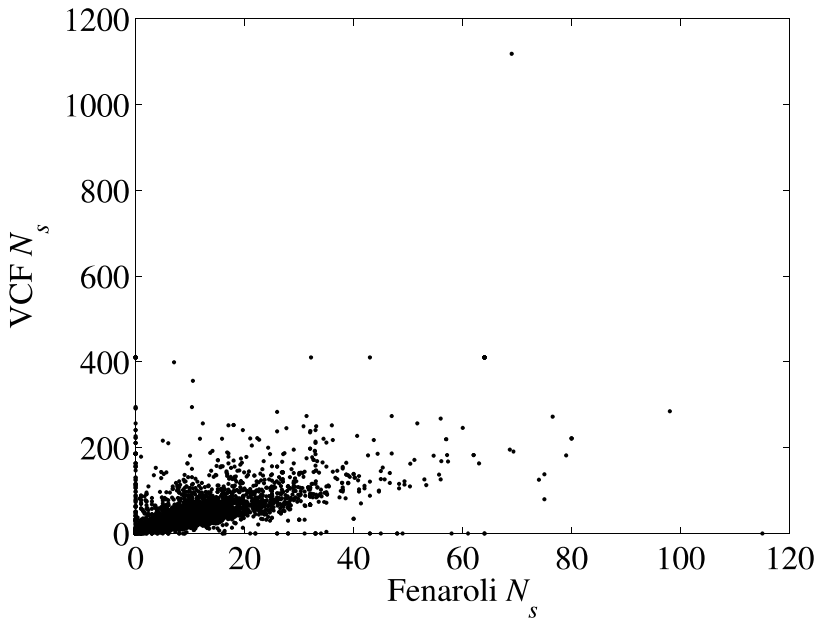

Figure 11: Scatter plot of the number of average shared flavor compounds per recipe calculated using the Fenaroli and VCF data sets.

novelty. Each domain of creativity, whether music, literature, or food recipes, needs a defined notion of quality. Flavor pairing is a putative quality metric for cooking, but our analysis here is not conclusive and indicates that there is more to the story.

Acknowledgments

The authors thank Nizar Lethif for assistance in preparing the VCF data set and Y.-Y. Ahn, Sebastian Ahnert, Cindi Avila, Debarun Bhattacharjya, James Briscione, Yi-Min Chee, Chris Gesualdi, Michael Laiskonis, Aleksandra Mojsilović, Florian Pinel, Angela Schörgendorfer, and Dan Stone for discussions.

References

[Ahn *et al.*, 2011] Yong-Yeol Ahn, Sebastian E. Ahnert, James P. Bagrow, and Albert-László Barabási. Flavor network and the principles of food pairing. *Scientific Reports*, 1:196, December 2011.

[Bhattacharjya *et al.*, 2012] Debarun Bhattacharjya, Lav R. Varshney, Florian Pinel, and Yi-Min Chee. Computational creativity: A two-attribute search technique. In *INFORMS Annual Meeting*, Phoenix, Arizona, October 2012.

[Blumenthal, 2008] Heston Blumenthal. *The Big Fat Duck Cookbook*. Bloomsbury, London, United Kingdom, 2008.

[Burdock, 2004] George A. Burdock. *Fenaroli's Handbook of Flavor Ingredients*. CRC Press, Boca Raton, Florida, 2004.

[Colton and Wiggins, 2012] Simon Colton and Geraint A. Wiggins. Computational creativity: The final frontier? In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 21–26, Montpellier, France, August 2012.

[Crosby, 1972] Alfred W. Crosby, Jr. *The Columbian Exchange: Biological and Cultural Consequences of 1492*. Greenwood Publishing, Westport, Connecticut, 1972.

[De Veaux and Hand, 2005] Richard D. De Veaux and David J. Hand. How to lie with bad data. *Statistical Science*, 20(3):231–238, August 2005.

[Guichard, 2002] Elisabeth Guichard. Interactions between flavor compounds and food ingredients and their influence on flavor perception. *Food Reviews International*, 18(1):49–70, 2002.

[Haddad *et al.*, 2010] Rafi Haddad, Abebe Medhanie, Yehudah Roth, David Harel, and Noam Sobel. Predicting odor pleasantness with an electronic nose. *PLoS Computational Biology*, 6(4):e1000740, April 2010.

[Khan *et al.*, 2007] Rehan M. Khan, Chung-Hay Luk, Adeen Flinker, Amit Aggarwal, Hadas Lapid, Rafi Haddad, and Noam Sobel. Predicting odor pleasantness from odorant structure: Pleasantness as a reflection of the physical world. *Journal of Neuroscience*, 27(37):10015–10023, September 2007.

[Kim *et al.*, 2003] Won Kim, Byoung-Ju Choi, Eui-Kyeong Hong, Soo-Kyung Kim, and Doheon Lee. A taxonomy of dirty data. *Data Mining and Knowledge Discovery*, 7(1):81–99, January 2003.

[King *et al.*, 2013] Silvia C. King, Herbert L. Meiselman, and B. Thomas Carr. Measuring emotions associated with foods: Important elements of questionnaire and test design. *Food Quality and Preference*, 28(1):8–16, April 2013.

[Kinouchi *et al.*, 2008] Osame Kinouchi, Rosa W. Diez-Garcia, Adriano J. Holanda, Pedro Zambianchi, and Antonio C. Roque. The non-equilibrium nature of culinary evolution. *New Journal of Physics*, 10:073020, 2008.

[Kort *et al.*, 2010] Miriam Kort, Ben Nijssen, Katja van Ingen-Visscher, and Jan Donders. Food pairing from the perspective of the 'Volatile Compounds in Food' database. In Imre Blank, Matthias Wüst, and Chahan Yeretzian, editors, *Expression of Multidisciplinary Flavour Science*, pages 589–592. Institut Für Chemie und Biologische Chemie, Zürich University of Applied Sciences, 2010.

[Lapid *et al.*, 2008] Hadas Lapid, David Harel, and Noam Sobel. Prediction models for the pleasantness of binary mixtures in olfaction. *Chemical Senses*, 33(7):599–609, September 2008.

[Lim and Johnson, 2012] Juyun Lim and Maxwell B. Johnson. The role of congruency in retronasal odor referral to the mouth. *Chemical Senses*, 37(6):515–522, July 2012.

[Morris *et al.*, 2012] Richard G. Morris, Scott H. Burton, Paul M. Bodily, and Dan Ventura. Soup over beans of pure joy: Culinary ruminations of an artificial chef. In *Proceedings of the International Conference on Computational Creativity*, pages 119–125, Dublin, Ireland, May–June 2012.

[Nunn and Qian, 2010] Nathan Nunn and Nancy Qian. The Columbian exchange: A history of disease, food, and ideas. *Journal of Economic Perspectives*, 24(2):163–188, Spring 2010.

[Shepherd, 2006] Gordon M. Shepherd. Smell images and the flavour system in the human brain. *Nature*, 444(7117):316–321, November 2006.

[Varshney *et al.*, 2013] Lav R. Varshney, Florian Pinel, Kush R. Varshney, Angela Schörgendorfer, and Yi-Min Chee. Cognition as a part of computational creativity. In *Proceedings of the 12th IEEE International Conference on Cognitive Informatics and Cognitive Computing*, New York, New York, July 2013.

[Veeramachaneni *et al.*, 2012] Kalyan Veeramachaneni, Ekaterina Vladislavleva, and Una-May O'Reilly. Knowledge mining sensory evaluation data: Genetic programming, statistical techniques, and swarm optimization. *Genetic Programming and Evolvable Machines*, 13(1):103–133, March 2012.

[Wang *et al.*, 2012] Dan J. Wang, Xiaolin Shi, Daniel A. McFarland, and Jure Leskovec. Measurement error in network data: A re-classification. *Social Networks*, 34(4):396–409, October 2012.

Learning recipe ingredient space using generative probabilistic models

Vladimir Nedović
Science Hub
Amsterdam, The Netherlands
vladimir.nedovic@science-hub.com

Abstract

In this paper, we demonstrate preliminary experiments using generative probabilistic models on recipe data. Recipes are reduced to lists of ingredients and analyzed in a bag-of-words fashion. We first visualize the highly-dimensional ingredient space and map it to different world cuisines. Latent Dirichlet Allocation (LDA) and Deep Belief Networks (DBN) are then used to learn generative models of ingredient distributions and produce some novel ingredient combinations. First results demonstrate the feasibility of the approach and point to its promise in recipe improvization.

1 Introduction

To the first approximation, a recipe consists of an ingredient list and the accompanying cooking instructions. In [Buykx and Petrie, 2011], the authors show that splitting recipe content into distinct blocks is rated best by the cooks who use the recipe. In addition, ingredient amounts are shown to be more useful within the method instructions than when presented together with the ingredient overview. Therefore, ingredient list and instruction sections can safely be addressed individually. In this work, we analyze the ingredient space of different recipes, resorting only to their ingredient lists.

A lot of research in the past several years focused on the recipe method, analyzing its text or augmenting its content using other modalities. We take a different approach. We observe that digital recipe texts, even in the ingredient overview part, are still rather static and isolated from each other regardless of similarities between the dishes they represent. Therefore, we aim to analyze ingredient lists at word level to provide an entry-point in recipe discovery.

Treating ingredients individually can lead to establishing ingredient correlations and recipe similarities. For instance, the former could allow for ingredient substitutions, whereas the latter could enable combining individual recipes for a single dish. In addition, dishes could be visualized in context, in terms of flavors combined, geographical region, and so on. For example, Wikipedia lists 28 different kinds of meatball dishes, even excluding the regional variations within countries. Given that all these dishes share the same ingredient base, they could easily be connected, and together give rise to a potentially new dish variant. In this paper, our goal is to demonstrate preliminary experiments which lead in that direction.

2 Related work

There has been an increasing body of research lately concerning automation related to recipe information and cooking activities. Beside many groups and companies taking part in a *Computer Cooking Contest*[1], workshops are being organized in conjuction with AI and multimedia conferences. In addition, we are seeing recommendation systems based on ingredient networks [Teng *et al.*, 2011] as well as attempts to quantitatively validate the food pairing hypothesis [Ahn *et al.*, 2011]. All these activities indicate an increasing interest in using computers in the kitchen, as well as increasing awareness of the importance of cooking in everyday lives.

Within the body of research, some approaches provide multimedia enhancements of textual recipe information. This is aimed at facilitating the process of cooking or at fixing some undesirable behavior [Wiegand *et al.*, 2012]. Whereas some methods focus on supporting a person while cooking [Hamada *et al.*, 2005; Ide *et al.*, 2010], others help people who follow a specific diet [Brown *et al.*, 2006; Chi *et al.*, 2007]. In contrast to these approaches, which augment existing textual information, we take a step back and concentrate on finding the limits of analysis of recipe texts, more specifically their ingredient lists.

The second group of recent approaches build on standard knowledge discovery (KDD) systems, adapted to recipe data. For example, the authors in [Gaillard *et al.*, 2012; Dufour-Lussier *et al.*, 2012; Mota and Agudo, 2012] focus on case-based reasoning (CBR) methods. CBR is the method of solving problems based on previous solutions to similar problems. However, this process mostly involves data ontologies and expert knowledge. We follow a different path. Instead of imposing structure on data, we aim to discover any structure that may be present in recipes using machine learning approaches.

We mostly draw inspiration from [Ahn *et al.*, 2011] and [Teng *et al.*, 2011]. However, where [Ahn *et al.*, 2011] analyzes foods at the level of flavor compounds, we limit ourselves to the level of individual ingredients. In contrast to

[1] `http://computercookingcontest.net/`

[Teng *et al.*, 2011], which computes complement and substitution networks based on co-occurences, we wish to go a step further and find longer-range ingredient and dish relationships as well. We aim to do so by employing appropriate pattern recognition methods.

In the machine learning literature, many document analysis methods rely on extensions of Latent Dirichlet Allocation (LDA) [Blei *et al.*, 2003]. LDA is a generative probabilistic model that represents each document as a mixture of a small number of hidden topics. In addition, each document word can be assigned to one of the document's topics. These methods can be easily applied to the recipe domain, as has been done in [Mori *et al.*, 2012], where the text of the recipe method is being analyzed. We also use LDA, but start by applying it to ingredient lists only, aiming to discover latent ingredient bases underlying specific recipes.

3 Generative Probabilistic Models

In this section, we give an overview of the two generative, latent variable models that we use for modeling ingredient distributions. Those are Latent Dirichlet Allocation (LDA) and Deep Belief Networks (DBN).

3.1 Latent Dirichlet Allocation

LDA is a generalization of the earlier Probabilistic Latent Semantic Analysis (PLSA) [Hofmann, 2001]. Both are well-known latent variable models for high dimensional count data, especially text in a bag-of-words representation. In this representation, D documents are each represented as a vector of counts with W components, where W is the number of words in the vocabulary. Each document j in the corpus is modeled as a mixture over K topics, and each topic k is a distribution over the vocabulary of W words. Each topic, ϕ_k, is drawn from a Dirichlet distribution with parameter η, while each document's mixture, θ_j, is sampled from a Dirichlet with parameter α. For each token i in the corpus, a topic assignment z_i is sampled from θ_{d_i}, and the specific word x_i is drawn from ϕ_{z_i}. The generative process is thus:

$$\theta_{k,j} \sim D[\alpha] \qquad \phi_{w,k} \sim D[\eta] \qquad z_i \sim \theta_{k,d_i} \qquad x_i \sim \phi_{w,z_i}.$$

Exact inference (i.e. computing the posterior probability over the hidden variables) for this model is intractable [Blei *et al.*, 2003] and thus a variety of approximate algorihtms have been developed. Ignoring α and η and treating θ_{kj} and ϕ_{wk} as parameters, we obtain the PLSA model, and maximum likelihood (ML) estimation over θ_{kj} and ϕ_{wk} directly corresponds to PLSA's Expectation-Maximization (EM) algorithm.

Starting from the form of log-likelihood,

$$l = \sum_i \log \sum_{z_i} P(x_i|z_i,\phi)P(z_i|d_i,\theta)$$

we obtain the parameter updates via standard EM derivation:

$$P(z_i|x_i,d_i) \propto P(x_i|z_i,\phi)P(z_i|d_i,\theta) \tag{1}$$

$$\phi_{w,k} \propto \sum_i \mathbb{I}[x_i = w, z_i = k]P(z_i|x_i,d_i) \tag{2}$$

$$\theta_{k,j} \propto \sum_i \mathbb{I}[z_i = k, d_i = j]P(z_i|x_i,d_i) \tag{3}$$

These updates can be rewritten by defining $\gamma_{wjk} = P(z = k|x = w, d = j)$, N_{wj} the number of observations for word type w in document j, $N_{wk} = \sum_j N_{wj}\gamma_{wjk}$, $N_{kj} = \sum_w N_{wj}\gamma_{wjk}$, $N_k = \sum_w N_{wk}$ and $N_j = \sum_k N_{kj}$. Then,

$$\phi_{w,k} \leftarrow N_{wk}/N_k \qquad \theta_{k,j} \leftarrow N_{kj}/N_j.$$

Plugging these expressions back into the one for the posterior in Equation 1, we arrive at the update,

$$\gamma_{wjk} \propto \frac{N_{wk}N_{kj}}{N_k} \tag{4}$$

where the constant N_j is absorbed into the normalization.

3.2 Deep Learning

Since no exact inference is possible in LDA and PLSA, they have to resort to slow or inaccurate approximations to compute the posterior distribution over topics. This makes it difficult to fit the models to data. In addition, there are limitations on the types of underlying structure that can be represented efficiently by a single layer of hidden variables.

To that end, Deep Belief Networks have been introduced in [Hinton *et al.*, 2006]. DBNs are generative probabilistic networks, composed of multiple layers of latent variables, which typically have binary values. They are usually represented by Restricted Boltzmann Machines (RBM) [Ackley *et al.*, 1985] at each layer. The layers of visible and hidden units are connected by a matrix of symmetrically weighted connections, optimized dring the learning phase.

Given an observed word count vector $\mathbf{v}$ and hidden topic features $\mathbf{h}$, let $\mathbf{v} \in \{1, ..., W\}^U$, where W is the dictionary size and U is the document size, and let $\mathbf{h} \in \{0, 1\}^F$ be binary stochastic hidden topic features. Let $\mathbf{V}$ be a $W \times U$ observed binary matrix with $v_u^w = 1$ if visible unit u takes on w^{th} value. The energy of the state $\{\mathbf{V}, \mathbf{h}\}$ is defined as follows:

$$E(\mathbf{V},\mathbf{h}) = -\sum_{u=1}^{U}\sum_{f=1}^{F}\sum_{w=1}^{W} M_{uf}^w h_f v_u^w - \sum_{u=1}^{U}\sum_{w=1}^{W} v_u^w b_u^w - \sum_{f=1}^{F} h_f a_f \tag{5}$$

where $\{M, a, b\}$ are the model parameters: M_{uf}^w is a symmetric interaction term between visible unit u that takes on values w, and hidden feature f; b_u^w is the bias of unit u that takes on value w, and a_f is the bias of hidden feature f. The probability that the model assigns to a visible binary matrix $\mathbf{V}$ is:

$$P(\mathbf{V}) = \frac{1}{Z}\sum_{\mathbf{h}} \exp(-E(\mathbf{V},\mathbf{h})), \tag{6}$$

where Z is the partition function or normalizing constant:

$$Z = \sum_{\mathbf{V}}\sum_{\mathbf{h}} \exp(-E(\mathbf{V},\mathbf{h})) \tag{7}$$

The conditional distributions are given by softmax and logistic functions:

$$p(v_u^w = 1|\mathbf{h}) = \frac{\exp(b_u^w + \sum_{f=1}^F h_f M_{uf}^w)}{\sum_{q=1}^W \exp(b_u^q + \sum_{f=1}^F h_f M_{uf}^q)} \tag{8}$$

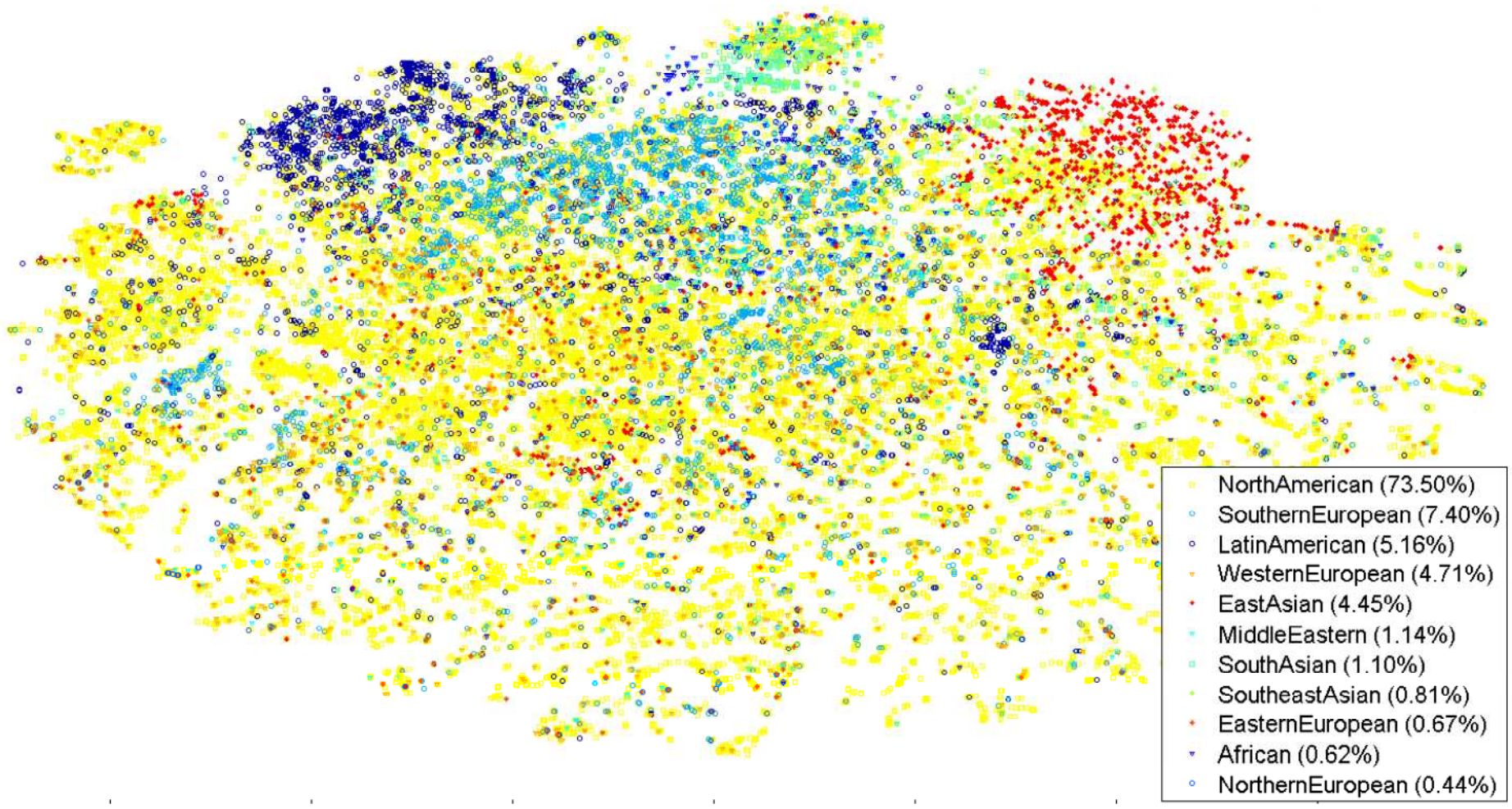

Figure 1: Visualization of ingredient space - a mapping to 2 dimensions using t-SNE. Different cuisines are represented by different markers. For each cuisine, the number in parentheses indicates the percentage of recipes that it covers in the dataset.

$$p(h_f = 1|\mathbf{V}) = \sigma\left(a_f + \sum_{u=1}^{U}\sum_{w=1}^{W} v_u^w M_{uf}^w\right). \qquad (9)$$

In document retrieval, a separate RBM is created for each document, and there are as many softmax units as there are words in the document. All the softmax units can share the same set of weights, connecting them to binary hidden units. Given a collection of N documents $\{\mathbf{V}_n\}_{n=1}^N$, the derivative of the log-likelihood w.r.t. parameters M takes the form:

$$\frac{1}{N}\sum_{n=1}^{N}\frac{\partial \log P(\mathbf{V}_n)}{\partial M_f^w} = \mathrm{E}_{P_{data}}[v^w h_f] - \mathrm{E}_{P_{model}}[v^w h_f], \qquad (10)$$

where $\mathrm{E}_{P_{data}}[\cdot]$ denotes an expectation w.r.t. the data distribution $P_{data}(\mathbf{h}, \mathbf{V}) = p(\mathbf{h}|\mathbf{V})P_{data}(\mathbf{V})$, with $P_{data}(\mathbf{V}) = \frac{1}{N}\sum_n \delta(\mathbf{V} - \mathbf{V}_n)$ representing the empirical distribution, and $\mathrm{E}_{P_{model}}[\cdot]$ is an expectation w.r.t. the distribution defined by the model. Exact maximum likelihood learning in this model is intractable, and learning is usually done by following an approximation to the gradient of a different objective function, a process called contrastive divergence.

4 Learning recipe ingredient space

4.1 Dataset

In this work, we aim to discover relationships, whether explicit or implicit, that may exist between different recipe vectors. For the preliminary experiments, we use the recipe collection of [Ahn *et al.*, 2011], which comes with more than 56000 recipes and 381 unique ingredients. The data was acquired by crawling three large recipe depositories, two American (*allrecipes.com*, *epicuirous.com*) and one Korean (*menupan.com*) (for parsing details, please see [Ahn *et al.*, 2011]). The recipes in the dataset are represented as ingredient lists only; therefore, we only consider the presence or absence of individual ingredients at this stage.

4.2 Cuisine mapping

To visualize the data and obtain some insight into its structure, we use the whole recipe corpus together with cuisine labels that are supplied with it. We utilize the technique of t-Distributed Stochastic Neighbor Embedding [van der Maaten and Hinton, 2008], which has shown promising results in visualizing the structure of high-dimensional data. Figure 1 shows a mapping of the ingredient space to different cuisines in two dimensions.

As can be seen from the figure, different cuisines are not equally represented in the dataset. In fact, North American recipes account for almost 3/4 of all the data. Nevertheless, certain conclusions can still be drawn even from this biased collection:

- South European, Latin American and East Asian recipes constitute very distinct dish groups;
- the above groups are nevertheless close to each other in the ingredient space;
- Asian cuisines (East, South and South East ones) are all connected into a separate cluster;

Figure 2: Latent Dirichlet Allocation. (a) Spatial visualization of 6 LDA topics. (b) Zoomed-in view of the two overlapping topics on the left, showing two different bases containing meat.

- Western European and North American groups cover the largest variety in ingredient space, overlapping with all other cuisines (although this effect is probably attributable to the bias in recipe distribution);
- there is a small group of Latin American dishes that is effectively closer to East Asian and South European cuisines than the Latin American one; same is true for a small group of South European dishes farther away from its base and closer to the North American tradition;
- Middle Eastern dishes sit between South European ones and East Asian ones;
- Eastern European dishes overlap with South European and North American ones; etc.

Therefore, different cuisines, which are essentially human constructs, indeed form distinct groupings in the ingredient space. This is observable, in only two dimensions, even when recipes are reduced to ingredient lists and only the ingredient presence/absence is considered.

4.3 Factor analysis

Latent variable models effectively project the data to a lower-dimensional manifold. However, the number of latent variables in such cases is not immediately apparent. Although many techniques exist for learning the intrinsic data dimensionality, most of them are very unstable, and we resort to an experiment with Factor analysis. The performance over different number of factors gives an indication of the number of latent variables for use in other models, such as LDA. For this purpose, we use the *Matlab Toolbox for Dimensionality Reduction* by Laurens van der Maaten[2].

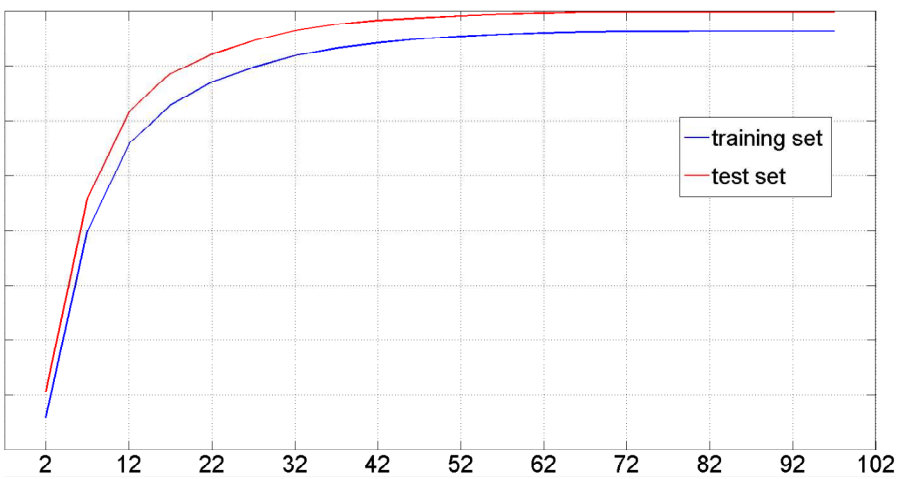

Figure 3: Factor analysis: log-likelihood vs. number of factors/dimensions used.

We split the data randomly into two halves, one used as the training set and the other as the evaluation set. For each different dimensionality, we learn the factor analysis mapping from the training data and compute the corresponding log-likelihood. That same mapping is then applied to the evaluation set, giving another log-likelihood value. This procedure is repeated for the number of factors ranging from 2 to 100, with a step of 5. The results are given in Figure 3, showing the log-likelihood leveling up after approximately 60 factors.

[2] http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html

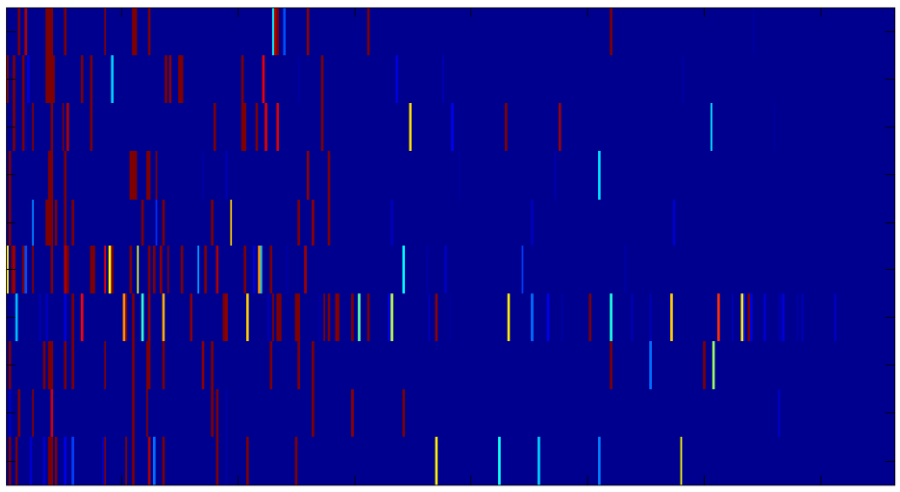

Figure 4: Ten new recipes sampled from the learned distribution given by a 3-level DBN.

4.4 Latent ingredient bases

Our goal is to analyze existing ingredient space and generate novel ingredient combinatons based on the learned models. We use LDA to project our highly-dimensional ingredient space to a smaller number of topics (i.e. *ingredient bases*) and then observe whether these bases make sense and how they are distributed. Since 60 or so dimensions, as given by factor analysis, would result in a cluttered figure, for visualization purposes we project to 6 topics only. The hyperparameters on the Dirichlet priors are set to $\alpha = 50/K$ and $\eta = 200/W$,: changing their values does not influence the results. We use the LDA package of [Griffiths and Steyvers, 2004] and show the results in Figure 2.

The figure shows the main ingredient bases maximally spaced apart on the horizontal axis: savory ones to the left and sweet ones to the right of the map. In addition to two "meat bases" on the left side, there is a base containing typical South-East Asian ingredients on top, and another one with common South European ingredients at the bottom of the figure. The ingredient base with common additives that affect acidity or fluidity of a dish is placed in the center of the map. The visualization implicitly shows the measure of how frequently certain (sets of) ingredients are used together.

4.5 Generating novel ingredient combinations

If we learn a generative probabilistic model from the ingredient data, we can also randomly sample it and observe the resulting ingredient combinations. These combinations will not necessarily correspond to those observed in the recipe corpus, but may represent completely novel varieties. In fact, one can imagine different parameter settings resulting in varying 'levels of combination novelty'.

For this purpose, we use a 3-layer DBN with an RBM at each layer. Beside the learning rate and the weight decay in learning the RBMs, which we fix at default values, the only additional parameters are the number of hidden nodes in each layer. In experiments presented here, we use 500 hidden variables in the first and second layer, and 2000 in the third. Changing the "network shape" by modifying these numbers can lead to somewhat different results, giving more exotic combinations, longer ingredient lists, etc.

We learn the network model and then sample the resulting distribution for 10 recipes. A color-map representation of these recipes is shown in Figure 4. In the figure, rows represent the 10 recipes, columns represent 381 ingredient dimensions, and color indicates probability of ingredient presence (blue = unlikely, red = very likely). Ingredients more to the left are those encountered earlier in the dataset, whereas more exotic ones are likely to be present at right. For example, the ingredient most used in the generated recipes, and visible as the connected strip at far left, is *butter*. Recipe contents are given in full in Table 4.5.

A quick inspection of the combinations in the table shows sensible pairings. Ingredients usually used for cakes are combined with fruits or nuts, whereas meat and seafood usually come with vegetables or herbs. Even some more unexpected combinations, e.g. involving fruits and vinegar, or fruits, meat and nuts, are in fact common in African cuisine. Another thing to note is that the number of ingredients is mostly between 10 and 20, out of the possible 381. These lists can be further constrained by increasing the threshold on the ingredient probability given by the network.

5 Conclusions

In this work, we demonstrate some preliminary experiments attempting to learn the ingredient space using machine learning approaches. To that end, we focus only on the ingredient list of each recipe and analyze it in a bag-of-words fashion. Beside exploiting this information to e.g. provide ingredient substitutions or obtain dish similarity networks, we aim to automatically generate novel ingredient combinations. We use Latent Dirichlet Allocation to learn and visualize latent ingredient bases, whereas novel ingredient combinations are generated using Deep Belief Networks. Preliminary results show the promise of this approach, resulting in sensible ingredient bases as well as novel ingredient combinations.

Future work will focus on the gathering of a less biased dataset and on recipe completion under the user-specified constraints. Extensions are likely to include ingredient amounts, as well as better visualization of ingredient and dish relationships.

References

[Ackley *et al.*, 1985] D. H. Ackley, G.E. Hinton, and T.J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

[Ahn *et al.*, 2011] Y.-Y. Ahn, S. E. Ahnert, J. P. Bagrow, and A.-L. Barabási. Flavor network and the principles of food pairing. *Scientific Reports*, 1(196), 2011.

[Blei *et al.*, 2003] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[Brown *et al.*, 2006] B. Brown, M. Chetty, A. Grimes, and E. Harmon. Reflecting on health: A system for students to monitor diet and exercise. In *Proceedings of CHI Extended Abstracts*, 2006.

[Buykx and Petrie, 2011] L. Buykx and H. Petrie. What cooks need from multimedia and textually enhanced

Table 1: Ingredient lists of newly generated recipes. Ingredients are sorted according to their prevalence in the whole dataset.

recipe	generated ingredient list
1	*egg, wheat, butter, milk, vegetable oil, cream, vanilla, cane molasses, corn, almond, pecan, coconut, tamarind, pineapple, cherry*
2	*egg, butter, onion, garlic, vegetable oil, black pepper, pepper, chicken, lemon juice, mushroom, parmesan cheese, thyme, green bell pepper, white wine, rosemary*
3	*butter, onion, garlic, olive oil, black pepper, vinegar, rice, cheese, parmesan cheese, macaroni, white wine, pea, squash, crab, asparagus*
4	*egg, wheat, butter, milk, vanilla, cinnamon, starch, almond, pecan, raisin, orange peel*
5	*egg, wheat, butter, vegetable oil, cinnamon, walnut, nutmeg, honey, apple, red wine, cranberry, orange peel, cardamom*
6	*wheat, butter, onion, garlic, black pepper, vinegar, carrot, chicken broth, rice, mushroom, soy sauce, lard, lemon, starch, almond, hazelnut*
7	*cream, vanilla, honey, almond, orange, pineapple, gelatin, mint, cranberry, grape juice, raspberry, apricot, rum, orange peel, peach, pear, brandy, plum, pistachio, berry*
8	*egg, wheat, butter, vanilla, cinnamon, walnut, nutmeg, honey, apple, almond, raisin, coconut, oat, nut, cherry, yogurt, cranberry, date*
9	*butter, vanilla, vinegar, cane molasses, lemon, walnut, apple, raisin, cider, fruit*
10	*egg, butter, vanilla, cinnamon, ginger, lemon, nutmeg, almond, coconut, orange, orange juice, apricot, cardamom*

recipes. In *IEEE International Symposium on Multimedia*, 2011.

[Chi *et al.*, 2007] P.-Y. Chi, J.-H. Chen, H.-H. Chu, and B.-Y. Chen. Enabling nutrition-aware cooking in a smart kitchen. In *Proceedings of CHI Extended Abstracts*, 2007.

[Dufour-Lussier *et al.*, 2012] V. Dufour-Lussier, F. Le Ber, J. Lieber, T. Meilender, and E. Nauer. Semi-automatic annotation process for procedural texts: An application on cooking recipes. In *Proceedings of the Cooking with Computers workshop (CwC)*, 2012.

[Gaillard *et al.*, 2012] E. Gaillard, E. Nauer, M. Levefre, and A. Cordier. Extracting generic cooking adaptation knowledge for the taaable case-based reasoning system. In *Proceedings of the Cooking with Computers workshop (CwC)*, 2012.

[Griffiths and Steyvers, 2004] T. Griffiths and M. Steyvers. Finding scientific topics. In *Proceedings of the National Academy of Sciences*, volume 101, pages 5228–5235, 2004.

[Hamada *et al.*, 2005] R. Hamada, J. Okabe, I. Ide, S. Satoh, S. Sakai, and H. Tanaka. Cooking navi: Assistant for daily cooking in kitchen. In *Proceedings of the annual ACM international conference on Multimedia*, 2005.

[Hinton *et al.*, 2006] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):15271554, 2006.

[Hofmann, 2001] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177–196, 2001.

[Ide *et al.*, 2010] I. Ide, Y. Shidochi, Y. Nakamura, D. Deguchi, T. Takahashi, and H. Murase. Multimedia supplementation to a cooking recipe text for facilitating its understanding to inexperienced users. In *Proceedings of the Workshop on Multimedia for Cooking and Eating Activities (CEA)*, 2010.

[Mori *et al.*, 2012] S. Mori, T. Sasada, Y. Yamakata, and K. Yoshino. A machine learning approach to recipe text processing. In *Proceedings of the Cooking with Computers workshop (CwC)*, 2012.

[Mota and Agudo, 2012] S. G. Mota and B. D. Agudo. Acook: Recipe adaptation using ontologies, case-based reasoning systems and knowledge discovery. In *Proceedings of the Cooking with Computers workshop (CwC)*, 2012.

[Teng *et al.*, 2011] C.-Y. Teng, Y.-R. Lin, and L. A. Adamic. Recipe recommendation using ingredient networks. In *Proc. 4th International Conference on Web Science*, 2011.

[van der Maaten and Hinton, 2008] L. J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, Nov 2008.

[Wiegand *et al.*, 2012] M. Wiegand, B. Roth, and D. Klakow. Knowledge acquisition with natural language processing in the food domain: Potential and challenges. In *Proceedings of the Cooking with Computers workshop (CwC)*, 2012.

Do Good Recipes Need Butter? Predicting User Ratings of Online Recipes

Ning Yu
University of Kentucky
Lexington, KY, USA
ning.yu@uky.edu

Desislava Zhekova
CIS, University of Munich
München, Germany
desi@cis.uni-muenchen.de

Can Liu
Indiana University
Bloomington, IN, USA
liucan@indiana.edu

Sandra Kübler
Indiana University
Bloomington, IN, USA
skuebler@indiana.edu

Abstract

In this work, we investigated the automatic prediction of user ratings for recipes. Information including the ingredients, the instructions, and the reviews from Epicurious were fed into a machine learner, a multi-class support vector machine, to examine how reliable they are when predicting recipe ratings. Our results show that information from the reviews results in the most reliable predictions: we reached an accuracy of 62%. The problem is difficult, partly because of the skewing of the ratings: most recipes are rated with 3 or 4 out of 4 forks.

1 Introduction

Exchanging recipes over the internet has become popular over the last decade. There are numerous sites that allow us to upload our own recipes, to search for and to download others, as well as to rate and review them. Such sites aggregate invaluable information, not only in terms of providing recipes, but also in providing information about cultural preferences with regard to food. For example, the site Epicurious[1] has more than 1 100 recipes for *chili*, but only 24 of those are *low sodium*. The site also presents 174 recipes for *muffins*, out of which 21 do not contain any dairy products. Such facts give us a first indication that Americans may eat more salty chilis than low-sodium ones, if we assume that the users of the site have a similar distribution as the American population. Additionally, a closer look reveals that out of all muffin recipes, only 11.5% have the highest rating of four forks[2], while among the non-dairy muffins, the percentage of four-fork ratings is 23.8%. From this, we could conclude that Americans like non-dairy muffins better than the ones containing dairy (keeping in mind the small size of the sample).

In this paper, we investigate whether we can predict user ratings for individual recipes, given the ingredients, the instructions, the reviews, or a combination of these features. If these experiments are successful, we can draw conclusions 1) about which ingredients good recipes include, 2) whether quantities of ingredients or 3) specific steps in the instructions have an influence on the ratings, and 4) whether we can detect reliable clues in the user reviews that allow us to deduce how much users like a recipe.

Questions 1) and 2) exploit intuitions and knowledge such as the inference that since fat is a flavor carrier, larger amounts of fat would generally increase taste and subsequently ratings. However, this does not carry across categories: having a larger amount of sugar in a cookie recipe may increase ratings while having the larger amount of sugar in a pot roast may have the opposite effect.

Question 3) is based on the assumption that easier recipes may be rated higher than more difficult ones. Difficulty may concern the number of steps in a recipe or involving certain techniques, such as melting chocolate in a double boiler rather than in a normal pot or measuring the temperature of melted sugar with a candy thermometer.

Question 4) is an extension of *sentiment analysis*, which investigates methods to classify users' attitudes towards a product (or other entities). In our case, the product is the recipe in question. However, the situation is complicated by the fact that we do not classify individual reviews or even sentences into positive or negative sentiment, but rather attempt to classify a recipe based on all reviews, which may be contradictory. The reviews for one randomly selected muffin recipe, for example, show ratings that range from 1 fork to 4.

The remainder of the paper is structured as follows: We will present related work in section 2. Then, we will discuss the data set and the questions that we will tackle in more detail in section 3 and the experimental setup in section 4. In section 5, we will present and discuss the results, and in section 6, we will conclude our findings and discuss future work.

2 Related Work

The history of applying computer technology to support cooking activities goes back to 1986 when CHEF, a case-based machine planner was created to generate new cooking plans from experience (i.e., old plans) [Hammond, 1986]. Later, various interactive cooking support systems including CounterActive [Ju *et al.*, 2001], eyeCook [Bradbury *et al.*, 2003], and Smart Kitchen [Hashimoto and Mori, 2008] were proposed. Recent developments focus on health driven cooking support systems [Karikome and Fujii, 2010; Kamieth *et al.*, 2011; Wagner *et al.*, 2011]. All these smart kitchen systems require heavy domain knowledge, which nowadays can

[1] `http://www.epicurious.com`

[2] Ratings in Epicurious are represented with 0 to 4 forks, including the intermediate values 1.5, 2.5 and 3.5.

be generated via crowd sourcing, e.g., online recipe sharing. In the last decade, researchers have studied online recipes for making recommendations to meet personal preferences [Ueda *et al.*, 2011] or health concerns [Freyne and Berkovsky, 2010; Mino and Kobayashi, 2009]. Such studies often focus on a specific cuisine or type of food, e.g., cookies, and need to build user models [Sobecki *et al.*, 2006]. Different from these recommendation systems, our work is content-driven and is interested in understanding the overall recipe preferences from all users. Nevertheless, recipe features used in such systems are applicable to our work.

Ingredients are the most commonly studied features for recipe recommendations. [Freyne and Berkovsky, 2010] considered all ingredients to be equally weighted within a recipe and aggregates the ratings of ingredients to predict the recipe rating. They found that this simple break down and construction approach worked better than a user-driven approach that takes user rating into consideration. While ingredient ratings are often not available, their experience shows the value of ingredients for predicting user ratings. Till today, most studies have treated ingredients equally [Forbes and Zhu, 2011; Teng *et al.*, 2012] and used them as binary features. [Zhang *et al.*, 2008] manually grouped ingredients into three levels of importance and ingredients that the researchers considered most important have the highest weight. To the best of our knowledge, this work is the first to use the actual quantities of ingredients within a recipe as feature values.

Cooking methods are another type of features that has been used for recipe recommendation. In the past, these features were either created manually [van Pinxteren *et al.*, 2011] or mined from existing knowledge bases [Teng *et al.*, 2012], which may be due to the noise in instruction text. Our work proposes a simple linguistic approach to directly extracting cooking methods and other features from instructions.

Although user reviews are the basis for recipes ratings, they have been used to identify refinements (e.g., reducing sugar in a recipe) [Druck and Pang, 2012; Teng *et al.*, 2012], but not to rate recipes. Our work uses sentiment analysis on recipe reviews and compares review features with content-based features in terms of their effectiveness for predicting ratings.

More recipe features can affect user ratings, and they are often used together. [van Pinxteren *et al.*, 2011] manually developed 55 features (e.g., soup, French cuisine), which covered 13 recipe characteristics (e.g., meal type, preparation time, preparation technique) for pasta. [Teng *et al.*, 2012] applied ingredient features, ingredient co-occurrence and substitution network features, and primary recipe features such as cooking methods, preparation time, and nutrition information to predict which recipe has the higher rating between a pair of similar recipes. They found that ingredient network features and nutrition features were most effective in their machine learning experiments, with accuracies of 75% and 78.6% respectively. When working with various features, [Freyne and Berkovsky, 2010; van Pinxteren *et al.*, 2011] used the weighted average to determine feature preferences, and [Forbes and Zhu, 2011] used a more sophisticated matrix factorization approach.

3 How to Predict User Ratings

In this section, we will first describe the data sets, and then the research questions that we are investigating in this paper.

3.1 Data Set

We developed a web crawler to scrape recipes with their information from the Epicurious site. Overall, we extracted more than 28 000 recipes. Then, we excluded all recipes for which we did not have the information to extract the features interesting to us (e.g., there are recipes that received no reviews and hence no rating information), which reduced the data set considerably, to 10 146 recipes. To avoid overwhelming the classifier by highly reviewed recipes, we did not include more than 10 reviews per recipe. For all recipes with more than 10 reviews, we took a random sample of 10 reviews out of the total number of reviews. We also only used full ratings, i.e., 3.5 forks are rounded down to the 3-fork class, based on the observation that users are generous when rating recipes. After this step, the data has the following distribution of classes/forks across all examples:

- 1 fork :112 examples
- 2 forks: 795 examples
- 3 forks: 5 670 examples
- 4 forks: 3 569 examples

This shows that we are dealing with a well known problem in machine learning: our data is heavily skewed towards higher rankings.

For every recipe, we extracted and prepared features in the following four groups:

- The overall rating (our gold standard classification) in terms of forks
- Metadata
  1. whether there is a picture
  2. whether there is wine pairing suggestion
  3. whether there is a quick meal label
  4. whether there is a healthy meal label
  5. whether it appears in the Epicurious menu
  6. the type of the recipe (e.g., Alcoholic, Salad)
  7. the type of cuisine (e.g., African, Italian)
  8. the recipe's dietary condition (e.g., Healthy, Vegan)
  9. number of available metadata items
  10. number of cuisine types associated with the recipe
  11. number of dietary conditions covered by the recipe
  12. the gender of the contributor
- Ingredients
  1. the ingredients used in the recipe that occur in ≥ 4 recipes
  2. the quantities of the ingredients
  3. the number of ingredients
  4. the main ingredients of the recipe (a separate group provided by Epicurious for each recipe)
- Instructions

1. the cooking steps/methods (e.g., chop, boil)
2. number of major steps (e.g., prepare the dough)
3. number of cooking steps

- Reviews
  1. a list of indicative uni-, bi-, and trigrams
  2. a list of best TF/IDF indicative uni-, bi-, and trigrams

Most features are represented by a binary value that indicates the presence or absence of each of the categories possible for this feature. For example, for each of the existing types of the recipe (e.g., Alcoholic, Salad, Sauce) we include a separate feature/value pair. Quantitative features, such as the number of ingredients, are represented with a scaled-down value ranging from 0 to 1 by dividing each feature value by the maximum number of features.

From the list of ingredients of a recipe, we extracted the ingredients and reduced them to their main nouns, noun compounds, or noun phrases. For example, the ingredient "3 tablespoons unsalted butter" is reduced to "butter", "1 Granny Smith apple, peeled, cored, and finely chopped (1 1/2 cups)" is reduced to the compound "Granny Smith apple". This was carried out based on a part-of-speech (POS) analysis of the ingredients. POS tagging is a well researched technique from Natural Language Processing that assigns a word class label to every word. Thus, both "Granny" and "Smith" would be assigned the label NNP, for proper noun, while apple is assigned NN, for common noun. The word "unsalted" would be assigned the label JJ for adjective. Excluding text within paraphrases, we kept all nouns (common and proper) and noun phrases that consist of one adjective modifier and one single noun head (e.g., "black pepper"). To avoid sparse features, we also excluded adjectives if the adjective falls into a manually created stopword list, including such words as "fresh". In order to use ingredient quantities as the values for ingredient features, we converted all the volume measures to tablespoon (tbsp). Thus, if one recipe needs "1 cup of sugar" while the other recipe needs "2 tablespoons of sugar", both instances will be assigned the feature "sugar_tbsp" and the values will be "16" and "2" respectively. We identified around 9 000 ingredient-measurement combinations and kept the ones that occur in at least 4 recipes to ensure that the features generalize well. Thus, we acquire a collection of 2 406 ingredient-measure features for our experiments.

For POS tagging, we used the Stanford POS tagger [Toutanova *et al.*, 2003][3], with the best performing model. However, since this POS tagger model was trained on the Wall Street Journal section of the Penn Treebank [Marcus *et al.*, 1993], we were using it out of domain, thus increasing the error rate. As a consequence, many verbs from the instructions are mistagged as nouns since they often occur as imperatives at the beginning of the sentence, and they occur more often as nouns in the Penn Treebank. For example, in the instruction "Drain over a rack.", the first word is mistagged as a noun instead of as a verb in the base form (VB).

From the instructions, we extracted all verb clusters and normalize them to the last verb. I.e., we extracted "set" from the sentence "You might want to set a timer." All verbs in past tense (e.g., "boiled", "chopped") and verbs that occur in less than 3 recipes were removed. Mistagged verbs were also manually removed. At the end, we had a total of 340 actions (e.g., "add", "roast", "stir").

From the reviews, we extracted n-grams (sequences of n words) that have discriminating capability between ratings. We consider uni-, bi-, and trigrams, based on words and on POS tags. E.g., "absolutely amazing", "very disappointing" etc. are covered by the bigram "RB JJ". "everyone loved" is based on "NN VBD". The most discriminating n-grams are the ones that achieved the highest term frequency/inverse document frequency (TF/IDF) scores. Since TF/IDF reflects how import a word is to a document, the higher the TF/IDF of a word, the more discriminating power it has. In calculating the TF/IDF values, we considered every review a document. TF/IDF values ranged from 6 to approximately 20 000. We then restricted the n-grams to the set of those that have a TF/IDF of at least 5 000.

3.2 Research Questions

In this work, we used a machine learning approach to investigate four main questions: 1) Can we reliably predict user ratings from the set of ingredients and their quantities? If this experiment is successful, we can conclude that there are specific ingredients or combinations of them that have an influence on how much users like the recipe. 2) Can we reliably predict user ratings from the instructions? In other words, are there certain instructions or combinations of them that influence user ratings, potentially because they are more complicated or difficult to execute? 3) Can we reliably predict user ratings from the set of reviews for the recipe? We assume that there is a close connection between the two, but the connection may be obscured by the fact that more people submitted ratings but only a small subset of these also provided reviews, which can vary considerably per recipe. Note that we approached all these questions indirectly, not by looking at statistical information directly but rather by employing classification. This has the advantage that we can use a successful classifier to predict the success for a novel recipe. This leads us to our last question: 4) Which types of information do we need in order to obtain a reliable rating classifier?

In order to investigate the first question concerning the ingredients and their quantities, we conducted the following experiments:

- INGREDIENT: Here, we used the ingredients and their quantities as well as the number of ingredients, i.e. features 1 through 3 from the Ingredients features in section 3.1. Ingredients are in the form of the ingredient plus the measurement for the quantity, e.g. sugar_tbsp; the quantities serve as values for their corresponding ingredient.
- INGR:NOQUANT: Here, we used the ingredients as presence features, i.e., they take binary values (present/not present) plus the number of ingredients.
- INGR:ADDMAIN: Here we used all features from INGREDIENT and also add the main ingredients, i.e., features 4 from the Ingredients features in section 3.1.

[3] http://nlp.stanford.edu/software/tagger.shtml

In order to investigate the second question, we conducted the following experiments:

- INSTRUCT: We used all features from the Instructions features in section 3.1.

In order to investigate question 3, we compare the following experiments:

- REVIEW:ALL: Here we used the full n-grams (without filter) from the Reviews group in section 3.1 (i.e. features 1-3).
- REVIEW:FILTER: In this setting, we only used the list of best TF/IDF indicative n-grams. (i.e. features 4-6).

The final question is concerned with finding the best classifier. We decided to experiment with the full set of features and with classifiers in which one of the feature sets is excluded:

- ALLFS: This setting includes all collected features.
- ALL:NOMETA: In this setting, all metadata features were removed.
- ALL:NOINGREDIENT: All features that represent the ingredients were excluded.
- ALL:NOINSTRUCT: All features that represent the instructions were excluded.
- ALL:NOREVIEW: In this setting, the set of all review features was excluded.

4 Experimental Setup and Evaluation

For classification, we used Support Vector Machines (SVMs) in the implementation of SVM^{light}. In particular, we employed the multi-class version SVM^{multiclass} [Crammer and Singer, 2002][4]. SVM is a machine learning classification approach which uses a function (called a kernel) to map example instances onto a linearly separable space.

All experiments reported here were performed with default parameter settings. In preliminary experiments, we observed that a higher value of the c parameter (i.e., the trade-off between training errors and margin) yields better results than its default value. However, it also resulted in unmanageable training time due to sheer volume of the data. In order to avoid benign data splitting, we used 10-fold cross-validation and report all results averaged over the 10 runs. As baseline, we used the setting in which the class with the highest number of recipes (the three-fork rating) is used to label all instances. We report accuracy for each evaluation setting as well as precision (P), recall (R), and F-scores (F) per class within the distinct settings. After feature extraction and composition of the feature vectors, we achieved a collection of 5 241 feature/value pairs that represent the information listed in four separate groups in section 3.1.

5 Experiments

5.1 How Important Are Ingredients and Their Quantities?

In order to investigate how important ingredients and their quantities are for user ratings, we carried out a classification experiment, in which we used the features from the ingredients (see section 3.1) and their quantities (INGREDIENT), and another one without quantities (INGR:NOQUANT). The baseline is shown first, and the results for the experiments with ingredients are shown in the second part of table 1.

The results show that we have a very competitive baseline: Just by choosing the class 3-forks for every recipe, we reached an accuracy of 56%. This is due to the heavy skewing in the distribution of the user ratings. When we used ingredient information, we reached a considerably lower accuracy of 50%. Almost none of the recipes were correctly classified as disliked recipes (1- and 2-forks). By leaving out the quantities (INGR:NOQUANT), we reached a marginally higher accuracy of 51%, gaining mostly in recall for the 3-fork rating. From these results, we can conclude that neither the list of ingredients nor the combination of these ingredients and their quantities are good predictors of ratings, and thus, it is unlikely that specific ingredients are directly related to the rating. Additionally, to answer the question in the title: While butter is a flavor carrier, it is the most frequent ingredient across all four ratings. Thus, butter is not a good discriminative feature.

In the next setting (INGR:ADDMAIN), we added the list of main ingredients as listed on the recipe page. This results in a higher accuracy of 55%, which is close to the baseline. However, it is interesting to see that we thus predicted a very similar distribution to the baseline: All recipes are grouped in the 3- and 4-fork categories, with a preference for the majority class of 3 forks. The results show that for the 3-forks rating, we still have a high recall (87%). This means, that the additional main ingredients help increase accuracy but at the expense of the lower ratings, meaning that the main ingredients are typical for 3-forked recipes, but not for poorly or highest rated recipes.

Sampling of ratings: Since we have a heavily skewed distribution of possible classes, we decided to apply up- and down-sampling to reach more balanced ratios of training instances. In downsampling, we restricted the number of training instances for the classes with more examples to match the number of instances in the least represented class (1 fork). In upsampling, the instances from lower classes in the training set are duplicated until they reach a more balanced ratio. For both methods, we tested different ratios. However, all these experiments resulted in a decrease in accuracy, so we used the full training set for the remaining experiments.

5.2 Can Instructions Predict Ratings?

For this question, we looked at the INSTRUCT setting in the third part of table 1. The results show that this setting is again close to the baseline and the INGR:ADDMAIN setting: We reached an accuracy of 56%, and all ratings are in the 3- and 4-fork category. From this, we can conclude that there is no direct interaction between the instructions and the distribution of ratings.

5.3 Can Reviews Predict Ratings?

For this question, we looked at the REVIEW:ALL and the REVIEW:FILTER settings in the fourth part of table 1. REVIEW:ALL uses the full list of n-grams from the reviews

[4] http://svmlight.joachims.org/svm_multiclass.html

		1-fork			2-fork			3-fork			4-fork			Acc.
		P	R	F	P	R	F	P	R	F	P	R	F	
1	BASELINE	0	0	0	0	0	0	0.56	1.00	**0.72**	0	0	0	0.56
2	INGREDIENT	0.02	0.05	0.03	0.10	0.03	0.05	0.57	0.71	0.63	0.39	0.28	0.32	0.50
	INGR:NOQUANT	0.03	0.05	0.03	0.14	0.02	0.04	0.57	0.74	0.64	0.40	0.28	0.33	0.51
	INGR:ADDMAIN	0	0	0	0.05	0	0	0.57	0.87	0.69	0.43	0.18	0.26	0.55
3	INSTRUCT	0	0	0	0	0	0	0.56	0.97	0.71	0.47	0.05	0.10	0.56
4	REVIEW:ALL	0.17	0.15	0.15	0.37	0.08	0.13	0.64	0.77	0.70	0.60	0.51	0.55	**0.62**
	REVIEW:FILTER	0.14	0.21	0.17	0.26	0.25	0.25	0.64	0.68	0.66	0.58	0.52	0.55	0.59
5	ALLFS	0.14	0.17	0.15	0.27	0.30	0.28	0.66	0.66	0.66	0.58	0.56	**0.57**	0.59
	ALL:NOMETA	0.18	0.21	0.19	0.28	0.29	0.29	0.66	0.68	0.67	0.59	0.55	**0.57**	0.60
	ALL:NOINGREDIENT	0.18	0.24	**0.21**	0.33	0.32	**0.32**	0.65	0.67	0.66	0.58	0.55	0.57	0.59
	ALL:NOINSTRUCT	0.14	0.18	0.15	0.27	0.30	0.28	0.66	0.67	0.66	0.58	0.55	**0.57**	0.59
	ALL:NOREVIEW	0	0	0	0.10	0	0	0.59	0.84	0.69	0.50	0.29	0.37	0.57

Table 1: The experimental results in all evaluation settings; P=precision, R=recall, F=F-score.

while for REVIEW:FILTER, we use the TF/IDF filtered n-grams. The results show that the REVIEW:ALL setting reaches the highest results with an accuracy of 62% and the best distribution over all ratings. When we filtered the n-grams, we reached higher recall and F-scores for the disliked categories: The F-score increases from 15% to 17% for the 1-fork rating and from 13% to 25% for 2 forks. However, this is offset by a loss in recall in the 3-fork rating, leading to a decrease in overall accuracy. This means that the filtered n-gram features lose their ability to distinguish 3-fork ratings to a certain extent. In the future, we will experiment with other feature selection methods.

5.4 Which Types of Information are Necessary?

For this question, we investigated whether we can obtain better ratings when we integrate information from different types of information about the recipe. Here, we looked at an experiment that used all available features, ALLFS, then we reduced the feature set by each group of features. The results are shown in the last part of table 1. Surprisingly, using all features results in an accuracy of 59%, which is higher than the baseline, but it is also considerably lower that the accuracy reached by using only review features.

The experiments where we removed groups of features show that only one group of features is detrimental to the overall results: the metadata features. The removal of this group results in a minimal improvement of accuracy over all features. This is not surprising since Epicurious attempts to promote all recipes. However, we will need a closer look to see whether individual features from this group may be useful. As expected, the removal of review features leads to the lowest accuracy (57%) in this group of experiments.

Overall, we can conclude that reviews provide the most reliable information for predicting ratings for recipes. However, if we aggregate all reviews into one decision, as we have done here, this means that the lower ratings are dispreferred by the classifier. From these results, we conclude that we need more information and more reliable features for the classifier.

6 Conclusion and Future Work

In this work, we investigated the prediction of recipes from the Epicurious site. We extracted information about the following categories: metadata, ingredients, instructions, and reviews. Our experiment showed that using only information from reviews results in the highest accuracy. However, this is a difficult problem because of the skewed distribution of the ratings. When using all n-grams from the reviews, we obtain an accuracy of 62%.

This work is still at the beginning. We are planning to improve the approach in three different areas: First, we will investigate our current features in more detail. Some features can be cleaned up further, and the groups of features we investigated may have been too coarse. I.e., looking at individual features may help us identify predictive features.

Second, we need to improve the approach to sentiment analysis with respect to the reviews. We are planning to integrate a separate classifier for individual reviews, so that we can include information about the distribution of the reviews into the recipe classifier. Here, we need to investigate which features are important for both classifiers in order to gain optimal results. We are also planning to explore the use of both general and corpus-based sentiment lexicons, e.g., Wilson's subjectivity terms [Wilson *et al.*, 2003] and a frequency lexicon containing patterns that could capture intentionally misspelled words (e.g., "luv," "hizzarious") [Yang *et al.*, 2008].

Third, we are planning to improve the Natural Language component and perform domain adaptation along the lines of [Kübler and Baucom, 2011]. We are also planning to integrate a dependency parser, MaltParser [Nivre *et al.*, 2007] into the system. The dependency parses provide syntactic information in form of dependencies (directed arcs) between pairs of words. The dependencies also carry grammatical information, which allows us to detect "who does what to whom". Additionally, they can also provide information with regard to the scope of negation. Negation has to be dealt with in sentiment analysis to ensure that "not good" is treated as a negative expression. However, more complex cases of negation such as " I don't think, the recipe is good." are often not handled correctly.

References

[Bradbury *et al.*, 2003] Jeremy S. Bradbury, Jeffrey S. Shell, and Craig B. Knowles. Hands on cooking: Towards an

attentive kitchen. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, pages 996–997, Fort Lauderdale, FL, 2003.

[Crammer and Singer, 2002] Koby Crammer and Yoram Singer. On the Algorithmic Implementation of Multiclass Kernel-Based Vector Machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

[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*, pages 545–553, Jeju, South Korea, 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*, pages 261–264, Hong Kong, China, 2011.

[Freyne and Berkovsky, 2010] Jill Freyne and Shlomo Berkovsky. Intelligent food planning: personalized recipe recommendation. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, pages 321–324, Santa Monica, CA, 2010.

[Hammond, 1986] Kristian J. Hammond. CHEF: A model of case-based planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 267–271, Philadelphia, PA, 1986.

[Hashimoto and Mori, 2008] Atsushi Hashimoto and Naoyuki Mori. Smart kitchen: A user centric cooking support system. In *Proceedings of the 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based System*, pages 848–854, Malaga, Spain, 2008.

[Ju *et al.*, 2001] Wendy Ju, Rebecca Hurwitz, Tilke Judd, and Bonny Lee. CounterActive: An interactive cookbook for the kitchen counter. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, pages 269–270, Seattle, WA, 2001.

[Kamieth *et al.*, 2011] Felix Kamieth, Andreas Braun, and Christian Schlehuber. Adaptive implicit interaction for healthy nutrition and food intake supervision. In *Proceedings of the 14th International Conference on Human-Computer Interaction*, pages 205–212, Orlando, FL, 2011.

[Karikome and Fujii, 2010] Shihono Karikome and Atsushi Fujii. A system for supporting dietary habits: planning menus and visualizing nutritional intake balance. In *Proceedings of the 4th International Conference on Ubiquitous Information Management and Communication*, pages 56:1–56:6, Siem Reap, Cambodia, 2010.

[Kübler and Baucom, 2011] Sandra Kübler and Eric Baucom. Fast domain adaptation for part of speech tagging for dialogues. In *Proceedings of the International Conference on Recent Advances in NLP*, Hissar, Bulgaria, 2011.

[Marcus *et al.*, 1993] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[Mino and Kobayashi, 2009] Yoko Mino and Ichiro Kobayashi. Recipe recommendation for a diet considering a user's schedule and the balance of nourishment. In *IEEE International Conference on Intelligent Computing and Intelligent Systems*, volume 3, pages 383–387, Shanghai, China, 2009.

[Nivre *et al.*, 2007] Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135, 2007.

[Sobecki *et al.*, 2006] Janusz Sobecki, Emilia Babiak, and Marta Slanina. Application of hybrid recommendation in web-based cooking assistant. In *Proceedings of the 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, pages 797–804, Bournemouth, UK, 2006.

[Teng *et al.*, 2012] Chun-Yuen Teng, Yu-Ru Lin, and Lada Adamic. Recipe recommendation using ingredient networks. In *Proceedings of the 3rd Annual ACM Web Science Conference*, pages 298–307, Evanston, IL, 2012.

[Toutanova *et al.*, 2003] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL 2003*, pages 252–259, Edmonton, Canada, 2003.

[Ueda *et al.*, 2011] Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima. User's food preference extraction for personalized cooking recipe recommendation. In *Proceedings of the Second Workshop on Semantic Personalized Information Management: Retrieval and Recommendation*, Bonn, Germany, 2011.

[van Pinxteren *et al.*, 2011] Youri van Pinxteren, Gijs Geleijnse, and Paul Kamsteeg. Deriving a recipe similarity measure for recommending healthful meals. In *Proceedings of the 16th International Conference on Intelligent User Interfaces*, pages 105–114, Palo Alto, CA, 2011.

[Wagner *et al.*, 2011] Juergen Wagner, Gijs Geleijnse, and Aart van Halteren. Guidance and support for healthy food preparation in an augmented kitchen. In *Proceedings of the 2011 Workshop on Context-awareness in Retrieval and Recommendation*, pages 47–50, Palo Alto, CA, 2011.

[Wilson *et al.*, 2003] T. Wilson, D. R. Pierce, and J. Wiebe. Identifying opinionated sentences. In *Proceedings of HLT-NAACL*, pages 33–34, Edmonton, Canada, 2003.

[Yang *et al.*, 2008] Kiduk Yang, Ning Yu, and Hui Zhang. WIDIT in TREC2007 blog track: Combining lexicon-based methods to detect opinionated blogs. In *Proceedings of the 16th Text Retrieval Conference*, Gaithersburg, MD, 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 *European Conference on Case-Based Reasoning*, pages 239–248, Trier, Germany, 2008.

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

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

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].

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

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	refreshing (salad)
most positive:	**most positive:**
Quinoa with Moroccan Winter Squash and Carrot Stew	Easy Cucumber Salad
Braised Provencal Chicken with Butternut Squash . . .	Fresh Fruit Salad with Honey, Mint and Lime Syrup
Spicy Sausage Soup with Cilantro	German Cucumber Salad with Sour Cream
Spicy Peanut Soup with Chicken	Mango Pineapple Salad with Mint
Italian Sausage and Tomato Soup	Cucumber Salad
most negative:	**most negative:**
Shrimp & Peppers Stir Fry	Warm Nut Encrusted Goat Cheese Salad with Bacon Lardons
Warm Jasmine Rice Salad with Shrimp and Thai Herbs	San Antonio Taco Salad
Pan-Fried Cod with Slaw	Creamy Potato Salad with Grilled Scallions
Garlic Bread Topped With Crab Meat and Spinach	Farmers' Market Salad with Spiced Goat Cheese Rounds
Singapore Chilli Prawns (Shrimp)	Sweet Corn and Basmati Rice Salad

picnic	chewy (cookies)
most positive:	**most positive:**
Macaroni Salad with Peas and Ham	Soft and Chewy Peanut Butter Cookies
Roast Beef Sandwiches with Lemon-Basil Mayonnaise	Chewy Chocolate Cookies I
Kittencal's Tuna Salad Sandwiches	Chewy Apple Oatmeal Cookies
My Family's Tuna-Pasta Salad	Chocolate Chewy Cookies
Radish Sandwiches	Soft Molasses Cookies
most negative:	**most negative:**
Potato Soup V	Pilgrim Hat Cookies
Smoky Four Cheese Macaroni Bake	Easy Cut-Out Cookies
Baked Spaghetti	Walnut Butter Cookies
The Best Butterscotch Banana Bread	Pill Bottle Cookies
Chicken with Vin Jaune and Morels	Chocolate Heart Cookies

creamy (soup)	spicy
most positive:	**most positive:**
Cream of Tomato Soup	Spicy Shrimp and Grits
Butternut Squash Soup	Grant's Famous Midnight Grill BBQ Sauce
Cream of Potato Soup III	Spicy Black Beans with Bell Peppers and Rice
Broccoli and Cheese Soup with Croutons	Spicy Filet Mignon
Creamy Potato Leek Soup	Black Beans and Tomatoes - Hot and Spicy
most negative:	**most negative:**
Vegetable and Ground Beef Soup	Spinach-Artichoke Ravioli-Lasagna
Vegetable-Sausage Soup	Perfect Chocolate Cake
Ham Bone Vegetable Soup 1967	Lemon Chiffon Pie with Gingersnap Crust
Wild Mushroom and Barley Soup	Goat Cheese and Onion Tarts
Beef Barley Soup	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 Després, 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.

Cooking Coach Spoken/Multimodal Dialogue Systems

Romain Laroche, Orange Labs, France, romain.laroche@orange.com
Jan Dziekan, Orange Labs, Poland, jan.dziekan@telekomunikacja.pl
Laurent Roussarie, Orange Labs, France, laurent.roussarie@orange.com
Piotr Baczyk, Orange Labs, Poland, piotr.baczyk@orange.com

Abstract

The use of a cookbook is quite awkward. Your working space is dirty, overloaded with kitchen utensils, your hands are probably busy, dirty or both and it is difficult for your eyes to keep track of the step you are working on. As a result, you keep going back and forth between the work surface and the cookbook, you lose time, track and may even forget a component as an unfortunate result of the mess.

Spoken dialogue systems are relevant for helping in a cooking task. Indeed, it can be performed without any device. This demonstration shows and describes Cooking Coach, two dialogue systems (one vocal and one multimodal) helping the user to search for a recipe, to check that she has all the ingredients and to prepare the recipe.

1 Spoken Dialogue System

Section 1 describes the spoken version of Cooking Coach. See Section 2 for the multimodal version.

1.1 Architecture

The architecture is illustrated by Figure 1. An Android client has been developed in order to use the tablet or mobile sound capture and restitution. The role of this client is to replace the Voice Platform that is commonly used for phone-based Interactive Response Services. We interface with the Dragon Mobile SDK, developed by Nuance, in order to get Voice Activity Detection, Automatic Speech Recognition and Text-To-Speech. On the other end, the dialogue application is deployed on an Application Server, which may have access to Information Systems if required.

A dialogue turn is typically processed as follows. Once Dragon Mobile has translated the audio signal into text, the Android client sends an HTTP request to the application server. An HTTP Server receives it and communicates its content to the Phase Engine's servlets. These latter asks the Semantic Analyser to interpret it and then processes the resulting request according to the dialogue logic. The servlets may load user libraries, which in turn, may have access to Information Systems. Once the servlets have planned the system's reaction, it calls a Java Server Page that generates the VoiceXML page to communicate to the client through the HTTP Server. The client parses the VoiceXML file, accesses the Nuance Server to synthesize the text (Text-To-Speech), plays the resulting sound to the user.

Figure 1: Architecture of the Cooking Coach dialogue system

1.2 Dialogue Example

S01: Welcome to CookingCoach. What recipe do you want to prepare?
U01: I would like to cook brownies.
S02: You want to cook brownies. For How many people?
U02: For two.
S03: To prepare walnut brownies, you need . . .
S04: Do you have all the ingredients?
U04: No.
S05: Which ingredient is missing?
U05: I don't have any walnut or chestnut.
S06: I can propose nutless brownies without your missing ingredients. Can you precise if you agree?
U06: Yes.
S07: Great, we will start the recipe.
S08: First, you'll need. . .
U08: Next/Repeat/Previous
. . .

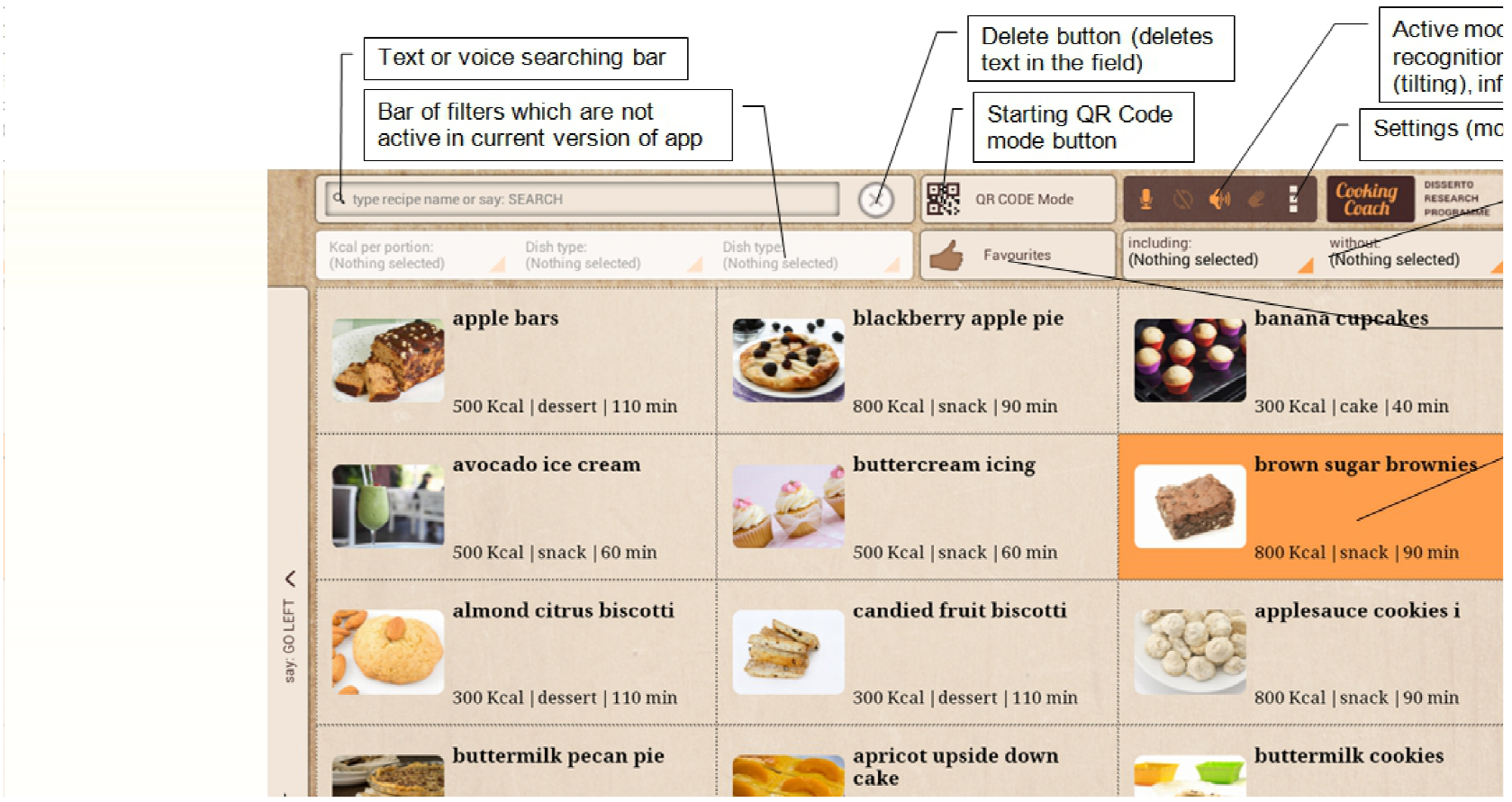

Figure 2: Multimodal Cooking Coach graphical user interface.

After the use names one or several missing ingredients, the system goes back to the recipe identification, and sends the same request forbidding these ingredients. The system remembers as well the number of people. Once the recipe has been validated, the system reads the steps one by one, with the possibility for the user to navigate through the steps : go further, repeat the current step and go back to the previous one.

1.3 Database Construction

For the purpose of the demonstrator, the database has been built with a website (www.allrecipes.com) hoover. The following fields are parsed:

- Name of the recipe
- Ingredients: list of quantity, unit and name of ingredients. We implemented an smart rounding algorithm to avoid to prepare 0.67 lemon.
- Number of persons: for how many people?
- Steps: steps to perform the recipe. We had to implement a step splitting method, based on punctuation and step size.
- A bunch of information not used in the application: rating, popularity, origin, poster...

2 Multimodal Cooking Coach

The multimodal Cooking Coach (see Figure 2) follows the same architecture and dialogue logic except if offers several additional multimodal ways to interact with the system:

- Touch screen : item selection / navigation button / ...
- QR Code : in order to select an ingredient you want in your recipe (filters the recipes with ingredients)
- Waving hand in front of the tablet : it enables to turn pages without touching your device which might be useful if your hands are dirty and the environment is too noisy to use speech recognition.

It also enables to cook several recipes in parallel.

Since a lot of effort has been made into the implementation of the multimodal prototype, we recorded two videos. The first one presents a use-case: `http://vimeo.com/62321504` and the second one shows the scope of the project with all its functionalities which have been done: `http://vimeo.com/61406788`. Password for both is: `muiDemo`.

The next steps of our prototype are to include a better search algorithm, in particular with filters amongst ingredients or type of food, and improve the personalisation of the application, *i.e.* avoid proposing recipes with ingredients that the user does not like.