# GoetheShaker* - Developing a Rating Score for Automated Evaluation of Cocktail Recipes

Marius Keppler **, Moritz Kohlhase, Niels Lauritzen, Matthias Schmidt, Pol Schumacher and Alexander Spät

Goethe University – Institute for Computer Science – Business Informatics,
Robert-Mayer-Straße 10, 60325 Frankfurt am Main, Germany
http://www.wi.informatik.uni-frankfurt.de

**Abstract.** In this paper we present a method for automated evaluation of cocktail recipes. Based on three components, we created a measure to assess the quality of a cocktail: harmony, popularity and profile match. The harmony represents how different ingredients harmonize with each other. The popularity helps frequently used ingredients stabilizing the recipe. The profile match ensures that cocktail recipes are well formed.

## 1 Introduction

Our paper presents our work for the mixology challenge of the 2014 Computer Cooking Contest (CCC)[1]. We satisfy the requirements by the creation of a functional CBR system, named GoetheShaker. CBR systems can be divided into four phases [1]. For the mixology challenge, we focussed on the latter two phases - revise and retain. The core of our paper is the creation of a suitable quality measure. This quality measure judges the retention of a recipe in the retain phase. With the focus on cocktail recipes, we build a quality measure, trying to reflect the interaction between the different cocktail ingredients. This function is made up from three components: Harmony, popularity and profile-match, which we will introduce in the following sections. The quality measurement uses statistical methods to identify relations in existing recipes. Additionally an ontology is used, to deduce inferences for newly mixed ingredients. We use the wikitaaable ontology, which contains information of around 2000 ingredients and is provided for this contest.
The paper is structured as follows. First, the general methodological fundamentals are explained. This is followed by the description of our implementation of the four CBR phases. Finally, we draw our conclusions, briefly discuss the testing of our prototype and point out suggestions for future work.

* This work was funded by the German Research Foundation, project number BE 1373/3-1.

** In alphabetical order.

[1] http://computercookingcontest.net

## 2 Methodology

In the following section, we introduce the fundamental methods used in our approach: The co-occurrence graph (CG), the pointwise mutual information (PMI), the Pagerank (PR), the Composition of Cocktails and the similarity measure.

*Co-occurrence graph* The CG represents the usage of ingredients within recipes. The nodes are ingredients used by the recipes of the CCC. We induce edges between every ingredient pair which occurs together in a recipe using PMI as edge-weight $h_{n,m}$. Teng et al. [12] created a similar network, they show that creating a CG and weighting edges with PMI allows estimating how well two ingredients complement each other. Therefore, the PMI can be used as an indicator of harmony.
The PMI is defined as a logarithm of quotient a posteriori and a priori probabilities[6] $PMI(a,b) = \log_2 \frac{p(a,b)}{p(a)p(b)}$ [12], it is "a measure of how much the actual probability of a particular co-occurrence of events $p(a,b)$ differs from what we would expect it to be on the basis of the probabilities of the individual events and the assumption of independence p(a)p(b)" [3]. $p(a,b)$ is the number of recipes containing ingredients $a$ and $b$ divided by the number of recipes; $p(a)$ and $p(b)$, the number of recipes containing ingredient $a$ or $b$ respectively, divided by the number of recipes. Since the PMI is symmetric [6], the CG is undirected.

Like Bouma [3] and Teng et al. [12] we normalized the PMI: $PMI \in [-1,1]$. As of now, the normalized PMI is meant when we speak of PMI. The PMI is 1 if two ingredients are perfectly correlated, that is when they only occur together. When the ingredients are distributed as expected under independence, the PMI is 0 [3]. PMI is -1 for ingredients that never occur together. Fig. 1 exemplifies a small part of our CG.

*The Pagerank* was developed by Lawrence Page and Sergey Brin. It is a method for node ranking in a linked database [10]. The idea is to rank a node very high, if other nodes are linked to it. The relevance of a node is determined by the relevance of its linked nodes which means that the significance of this node is defined recursively by the significance of the other nodes. The PR concept was initially introduced for ranking websites.

*Composition of Cocktails* In order to evaluate the composition of cocktails we need to classify every ingredient regarding five dimensions.

According to Cheng-Chang et al. [4] we distinguish between: Strong ($S_1$), Sweet ($S_2$), Sour ($S_3$), Weak ($W$) and Bitters ($B$). The dimension "Strong" describes the alcohol concentration, whereas "Sweet" and "Sour" are measured by the concentration of sugar and ascorbic acid respectively. Finally, "Weak" is the share of water and "Bitters" are the number of components that bring in flavor (like spices, flavor concentrate and bitter substances) or color in the cocktails.

We used the ingredient attributes from the ontology to setup the values of the different dimensions. As result, every ingredient $i$ has its own vector:

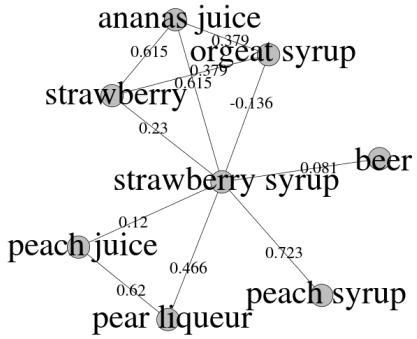


**Fig. 1.** A small community in the CG with PMI as edge weight.

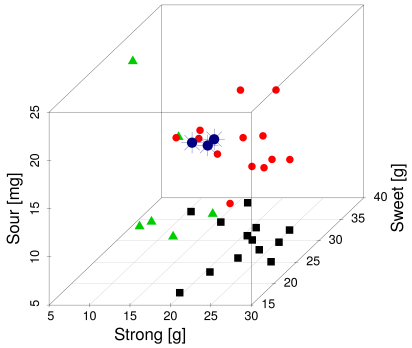


**Fig. 2.** Sample cluster of cocktails mapped into tree dimensions strong, sweet and sour. Cluster centers are represented by the bigger blue points.

$c_i = (S_1, S_2, S_3, W, B)$, in which the magnitude of the dimensions is defined (e.g. whether an ingredient is sweet and if so, how sweet, depending on its share of sugar). All ingredient vectors of one recipe can be added to classify a certain cocktail $r_j$ within the defined five dimensions: $c_{r_j} = \sum_{i \in I_{r_j}} c_i$

$I_{r_j}$ is the set of ingredients in $r_J$ and $c_{r_j}$ is the cocktail profile (COP) for $r_j$. This enables a cluster analysis of the data, which allows to identify major cocktail categories (figure 2). A simple example is the category of alcohol free cocktails, in which every cocktail has a zero share of the $S_1$ factor. Each cocktail category is symbolized by one cluster and its profiled by the center of it's cluster. A profile of a cocktail category is named category profile (CAP) and is represented by the vector $c_{category} \in C_{categories}$ (e.g. $c_{alcoholfree}$), where $C_{categories}$ is the set of all CAPs. The cluster analysis is done by the standard K-Means algorithm.

*Similarity measure* For measuring similarity between two ingredients $i$ and $j$ we use an ontology based similarity measure $Sim(i, j)$, like the measure proposed by Lin [7] in combination with Resnik's [11] self-information function.

## 3 GoetheShaker

Even if our work focuses on evaluating of CBR systems and therefore the third stage of the CBR cycle by Aamodt and Plaza [1], a proper implementation of the retrieve and reuse phases is still necessary. The task of the retrieve phase is the selection of one or more cases from the case base, which are appropriate to a given user query [2]. After a case has been retrieved, this case has to be adapted to the given situation in the reuse phase.

### 3.1 Retrieve

The core of our retrieve and reuse algorithm is a bipartite graph which contains recipes and ingredients as nodes, where each recipe is connected with its used ingredients. If a user submits a query, a set of wanted ingredients as well as unwanted ingredients is specified. Therefore we use two types of edges in the graph: Edges connecting recipes with wanted ingredients on the one hand and edges connecting unwanted ingredients on the other. The recipe node with the most edges of the wanted type is the recipe to adapt in reuse. If two or more recipes have the same amount of wanted ingredients, the recipe with less unwanted ingredients is chosen. If they have the same amount of unwanted ingredients, a random selection is made.

### 3.2 Reuse

The retrieved recipe is the candidate for adaption. If all wanted and no unwanted ingredients specified in the query are part of the recipe, the reuse phase as well as all other phases of the CBR cycle are completed. A user query contains a set of wanted and unwanted ingredients. The retrieved recipe contains additional, neither wanted, nor unwanted ingredients, which are called optional ingredients. These optional ingredients are combined with the unwanted ingredients as candidates for substitution. In the next step as many optional and unwanted ingredients as needed or as possible are substituted with missing remaining wanted ingredients. Based on pairwise comparison using $Sim(i, j)$, we determine the best (most similar) candidates $i, j$ for substitution. One by one we replace unwanted or optional ingredients with wanted ones. To avoid replacing ingredients with dissimilar other ingredients we use a minimum threshold for substitution. Several cases can emerge from this scenario, from which only the final one needs a special treatment:

- All wanted ingredients are now part of the recipe and no optional ingredients remain.
- Some optional or unwanted ingredients remain beside the wanted ones.
- Some wanted ingredients are still not in the recipe, but no unwanted or optional ingredients are left for substitution. In this case, the remaining missing ingredients are added.

Lastly, as no unwanted ingredients should remain in the recipe, all remaining unwanted ingredients are replaced by the most similar not unwanted ingredients, or deleted if there exists no appropriate substitution.

### 3.3 Revise

The third phase of CBR, named Revise, evaluates the adapted case and offers a quality measurement to assess whether this case should be retained. In the context of mixing cocktails, we assess the quality of a cocktail recipe. To enable the automatic evaluation of cocktail recipes, criteria for good cocktails are needed. GoetheShaker uses three criteria to rate the quality of a cocktail recipe:

- **Harmony:** How the ingredients of one cocktail harmonize. There exists pairs that harmonize well, like lemon juice and gin, whereas other combinations don not harmonize. In a good cocktail recipe each ingredient blends well with the other ingredients.
- **Popularity:** Describes the popularity of the ingredients. Popular ingredients are used in many recipes and can be mixed with a lot of other ingredients.
- **Profile-Match:** A cocktail has to be well-defined. As mentioned above, the profile of a cocktail can be used to map the cocktail into five dimensions. Thus, typical sorts of cocktails, e.g. "Sour" or "Alcohol-free", have their own category profiles. An indicator that a cocktail is well formed is that its profile is close to one of the category profiles.

**Harmony** The calculation is based on the CG. For a specific cocktail recipe the harmony $H_{r_j}$ is calculated by the average of the harmonies $h_{n,m}$ between all ingredients $n, m$ for the adopted recipe:

$$H_{r_j} = \frac{avg(\{h_{n,m} | n, m \in I_{r_j}, n \neq m\}) + 1}{2} \tag{1}$$

Beside the average we add 1 and divide it by 2 for scaling the harmony into the same range ($[0, 1]$) as the other components of our quality measure. Suppose, if a cocktail has three ingredients: Vodka, strawberry and lime juice. The harmony of this cocktail will be calculated from this matrix:

| | vodka | strawberry | lime juice |
|---|---|---|---|
| vodka | | $h_{vodka,strawberry}$ | $h_{vodka,lime\ juice}$ |
| strawberry | | | $h_{strawberry,lime\ juice}$ |
| lime juice | | | |

The harmony between two ingredients is their $PMI$ (e.g. $h_{vodka,strawberry}$) that is the edge weight in the CG between both nodes.

But the PMI between two ingredients only exists if these ingredients have ever been mixed before. In this case, a non-existing $PMI$ could be set to the worst possible value (-1) in order to punish ingredients-pairs that has never been mixed. Hence, as an essential disadvantage, we do not distinguish between two ingredients which can potentially be mixed in the future (e.g. vodka and strawberry) and ingredients which in fact do not harmonize (e.g. vodka and tuna). Because of that, solutions to estimate the harmony between two ingredients that have never been mixed before have to be found and evaluated.

The problem could be framed as a path finding problem. The harmony between two ingredients would then be the average PMI along the path that connect the two ingredient nodes. However, obviously transitivity does not apply in the CG. For example a CG could have three nodes *pepper*, *vodka* and *orange juice*. Because of the "Bloody Mary" cocktail *pepper* and *vodka* are connected and and because of the "Sex on the Beach" cocktail *vodka* and *orange juice* are connected. If transitivity would apply, it would be possible to conclude that *pepper* and *orange juice* fit well together which is not the case.

We decided to use $Sim(i, j)$ to substitute missing edges or PMI values respectively with the PMI of similar ingredient combinations between which edges exist. Therefore, the procedure is as follows: First, a general threshold is defined as a search radius for substitutes because we want to replace ingredients with similar other ingredients like lemons with citrons. But we do not want inappropriate substitutions, like lemons with tuna. Second, we search until we have found an ingredient-combination that is an edge in our CG, starting with the most similar ingredient. We then use the $PMI$ of the newly found ingredient combination ($PMI^*$) for calculating the harmony. If no appropriate substitutes are found within the search radius, the worst possible PMI ($-1$) will be used. This is justified because not only have the two original ingredients never appeared together in one recipe of the case base, but also the combination of similar ingredients has not. To take into account that we do not use the original PMI values but the ones from the substitutes, we multiply the $PMI^*$ with a term $0 < t < 1$ to adjust the impact of $PMI^*$ concerning the harmony.

**Popularity** The problem with using the PMI for harmony is that ingredients which are used very frequently (in many recipes) correlate poorly with other ingredients. Thus, those ingredients do not fit other ones accordingly, even though they usually harmonize well with other ingredients. Hence, popularity ($P$) has been introduced to correct this (as a stabilizer). It is calculated by using the average of the PRs of all ingredients of a recipe because, as mentioned in the paragraph about the PR, the more other ingredients an ingredient is connected with, the higher is its PR. Finally, a high PR implies a high $P$. For calculation reasons, the PR is normalized in $\in [0, 1]$. The normalized PR of ingredient $i$ is called $PR_i^n$. Thus for a certain recipe $r_j$:

$$P_{r_j} = avg(\{PR_i^n | i \in I_{r_j}\}) \tag{2}$$

**Profile-Match** The Profile-Match(PM) is a measure of how well-formed a cocktail is. The basis for this calculation are the COP and CAP vectors as described in section 2. The PM is the distance between the COP and the its nearest CAP. Because the further a COP is away from the CAPs, the more the composition of its underlying cocktail differs from conventional cocktail compositions. Figure 2 illustrates (but only for three dimensions) three simple clusters (built from sample random recipe data) with its CAPs marked as large blue points. The distance between the COP and a CAP is measured using the Euclidean distance ($d$) normalized to the range $[0, 1]$. The PM for recipe $r_j$ is then calculated as follows:

$$PM_{r_j} = 1 - min\{d(m, n) | m = c_{r_j}, n \in C_{categories}\} \tag{3}$$

Finally to calculate the quality for a certain recipe $r_j$, we determine a weighted average of the harmony ($H_{r_j}$) and the popularity ($P_{r_j}$) and adjust this average by the profile match ($PM_{r_j}$):

$$Q_{r_j} = PM_{r_j} * \left( \frac{\alpha * H_{r_j} + \beta * P_{rj}}{2} \right) \tag{4}^1$$

The mixology challenge requires the system to provide a quality estimation from 0 (horrible) to 5 (fantastic). Therefore we map our measure on this scale [2].

### 3.4 Retain

The retain phase is the learning phase of a CBR system. In terms of the utility function provided by Lopez de Mantaras et al.[9], GoetheShaker uses the $Q$ from the revise phase to decide whether the case should be added to the case base. For this decision we use a threshold $q$, which is defined as the median of the quality of all recipes in the case base. We will store the recipe in the case base, if its quality $Q$ is higher than the threshold $q$. If the case is retained, the bipartite graph, CG, PR and CAP will be updated with the new data from the case base.

## 4 Conclusions

At this time our prototype is not yet completed. As soon as we have a functional system, we will evaluate our methods by testing it with domain experts. These experts can provide us with a proper assesement of our system and especially of the three components of the revise phase. Our goal is to verify the significance of each component concerning the cocktail quality.

Our approach is mainly based on statistical methods, e.g. the measurement of occurrences and thus, strongly based on sample size and the quality of a sample. In our case we used small samples of wikitaaable which includes about 100 cocktails. Larger data sets with cocktail recipes provide a greater sample size, but the data quality is often worse. The data contains brand names and duplicates that would have to be manually mapped to the ingredients in the ontology we use. Another disadvantage is the diminishing meaningfulness of the PR with growing data sets. With a growing case base, our CG matures from a random graph to a small-world network (or even to a full graph). In a full graph all nodes are very important and the significance of the PR converges to zero. Another improvement for finding better cocktail categories could be a variation of the K-Means clustering algorithm.

In addition we want to point out the method of determing the harmony between newly mixed ingredients by using the harmony of similar ingredients derived through the ontology. This method enables us to make intelligent assumptions about the harmony of ingredient pairs that have never been mixed before. Beyond this, a chemical analysis[3] could be conducted for assessing the quality of newly mixed ingredients pairs.

Future work can consider the use of our revise algorithm to improve the retrieve and reuse phase, also the adaption of GoetheShaker to other areas like general food recipes could be evaluated.

[1] The parameters $\alpha$ and $\beta$ are used to adjust $Q$ to the prototype and thus can not be explicitly defined until the prototype is completed.

[2] The detailed mapping can not be devised until the prototype is completed.

[3] See `https://www.foodpairing.com` (Last visited 23.06.2014)

## References

[1] Agnar Aamodt and Enric Plaza. "Case-based reasoning; Foundational issues, methodological variations, and system approaches". In: *AI COMMUNICATIONS* 7.1 (1994), pp. 39–59.

[2] Ralph Bergmann et al. "Case-Based Reasoning – Introduction and Recent Developments". In: *KI - Künstliche Intelligenz, German Journal on Artificial Intelligence - Organ des Fachbereiches "Künstliche Intelligenz" der Gesellschaft für Informatik e.V. (KI)* 23.1 (Jan. 2009), pp. 5–11.

[3] Gerlof Bouma. "Normalized (pointwise) mutual information in collocation extraction". In: *Proceedings of GSCL* (2009), pp. 31–40.

[4] Cheng-Chang et al. "Evaluation Teaching Cocktail with the Instructional Module of Cocktail". In: *International Business Management* 4.2 (2010), pp. 57–66.

[5] Church, Kenneth Ward, and Patrick Hanks. "Word Association Norms, Mutual Information, and Lexicography". In: *Comput. Linguist.* 16.1 (Mar. 1990), pp. 22–29. ISSN: 0891-2017.

[6] Robert Mario Fano. *Informationsübertragung : eine statistische Theorie der Nachrichtenübertragung*. München [u.a.]: Oldenbourg, 1966.

[7] Dekang Lin. "An information-theoretic definition of similarity." In: *ICML* 98 (1998), pp. 296–304.

[8] Beatriz López. *Case-Based Reasoning - A Concise Introduction*. Morgan & Claypool Publishers, 2013. ISBN: 978-1-627-05008-1.

[9] Ramon Lopez de Mantaras et al. "Retrieval, reuse, revision and retention in case-based reasoning". In: *The Knowledge Engineering Review* 20 (03 Sept. 2005), pp. 215–240. ISSN: 1469-8005. DOI: 10.1017/S0269888906000646.

[10] Lawrence Page et al. "The PageRank citation ranking: Bringing order to the web." In: (1999).

[11] Philip Resnik. "Using Information Content to Evaluate Semantic Similarity in a Taxonomy". In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*. IJCAI'95 (1995), pp. 448–453.

[12] ChunYuen Teng, Yu-Ru Lin, and Lada A. Adamic. "Recipe recommendation using ingredient networks". In: *CoRR* abs/1111.3919 (2011).

[13] Volker Turau. *Algorithmische Graphentheorie*. Oldenbourg, 2009. ISBN: 9783486590579.