Overcoming the Long Tail Problem: A Case Study on CO2-Footprint Estimation of Recipes using Information Retrieval

Melanie Geiger[1,2], **Martin Braschler**[2]
[1] Université de Neuchâtel, Neuchâtel, Switzerland
[2] Zurich University of Applied Sciences, Winterthur, Switzerland
{imhf, bram}@zhaw.ch

Abstract

We propose approaches that use information retrieval methods for the automatic calculation of CO2-footprints of cooking recipes. A particular challenge is the "long tail problem" that arises with the large diversity of possible ingredients. The proposed approaches are generalizable to other use cases in which a numerical value for semi-structured items has to be calculated, for example, the calculation of the insurance value of a property based on a real estate listing. Our first approach, ingredient matching, calculates the CO2-footprint based on the ingredient descriptions that are matched to food products in a language resource and therefore suffers from the long tail problem. On the other hand, our second approach directly uses the recipe to estimate the CO2-value based on its closest neighbor using an adapted version of the BM25 weighting scheme. Furthermore, we combine these two approaches in order to achieve a more reliable estimate. Our experiments show that the automatically calculated CO2-value estimates lie within an acceptable range compared to the manually calculated values. Therefore, the costs of the calculation of the CO2-footprints can be reduced dramatically by using the automatic approaches. This helps to make the information available to a large audience in order to increase the awareness and transparency of the environmental impact of food consumption.

Keywords: BM25 weighting scheme adaptation, cooking recipe retrieval, CO2-footprint estimation

1. Introduction

One easily measurable quantitative quality criterion of a language resource (LR) is its coverage. However, achieving a high coverage usually requires a lot of human effort. One of the reasons is that often the frequencies of potential LR entries; e.g. words in human language or food products in cooking recipes (Müller et al., 2012), arrange themselves according to the so-called Zipf's law (Zipf, 1949), meaning that most entries relate to entities that occur very infrequently. Therefore, LRs are most likely never complete. This long tail problem is relevant for most applications that rely on LRs, however, it is particularly severe for information retrieval (IR) applications that not only use the LR to enhance their effectiveness (e.g. expanding queries with synonyms) but directly use the LR entries to compile their output.

To illustrate, consider a new class of retrieval applications that require the calculation of a single numerical value from a semi-structured item that consists of a list of textual elements. For example, such a semi-structured item may be a cooking recipe in which the elements are the instruction lines, such as "100g carrots, sliced" and "1 pizza dough", or a real estate listing in which the elements are the components, such as "Bedrooms: 4" and "Heating: Oil-Fired Central Heating". In those examples the numerical values to be calculated can be the nutrition value or the CO2-footprint of a recipe or for the real estate example the insurance value of a property.

An *element-wise* approach to calculating such values splits the problem into sub-problems by first calculating the value for each element individually and then computing the value of the complete item by aggregating the values of the individual elements. For most use cases, this means that the individual elements are matched to an LR, which then helps to estimate their values. In the case of recipes, the LR (Eaternity AG, 2017) we use contains the nutrition value and the CO2-value for each food product. For the estimation of real estate insurance values, a suitable LR contains the costs of the corresponding components; e.g. the average costs of a bathroom with a shower and a double washbasin. This *element-wise* approach, however, heavily relies on the completeness of the LR and has to use a fallback strategy if elements are not found in the LR. In practice, the fallback usually means that additional entries need to be added manually, an excessively costly option.

In the real estate example, an alternative human line of action is often to estimate the value of a property based on the values of other similar properties for which the value is already known. Hence, the value is estimated based on the whole item rather than the individual elements and thus the problem of the incompleteness of the LR can be circumvented. Gonzalez and Laureano-Ortiz (1992) replicate this process for automatic property appraisal. We propose an *item-based* approach using IR technology. We claim that this approach is applicable to many scenarios that include the calculation of a value for a semi-structured item whenever a similarity between the items can be defined.

In this paper, we focus on the use case of the automatic calculation of CO2-footprints of cooking recipes. The motivation for such a use case is that about one-third of CO2-emissions produced by the final household demand in Europe is caused by the consumption of food (Tukker and Jansen, 2006) and that the calculation of CO2-footprints for cooking recipes helps to increase the awareness and transparency of the environmental impact of food consumption. However, so far the footprint of a recipe was calculated with a manual process (O'Connor et al., 2018) which is time-consuming and therefore too costly to be applied to a wide range of cooking recipes.

We describe and evaluate an *element-wise*, an *item-based*, and a *hybrid* approach, combining the two, to automatically calculate the CO2-footprints of recipes. In the context of our CO2 use case, we call the *element-wise* approach "ingredient matching" and the *item-based* approach "recipe matching". The ingredient matching approach uses an IR pipeline to match the instruction lines to the corresponding entries in the LR through retrieval from an index. The recipe matching approach finds the most similar recipe in a corpus of indexed recipes for which the CO2-footprints are already assessed. A novelty is our proposal of an adapted version of the BM25 weighting scheme which also considers the amounts of the individual ingredients in the recipes. Finally, the hybrid approach combines the two other approaches so that a higher accuracy and stability of the CO2-value estimates can be achieved.

In our experiments, we compare the automatic approaches to the manual process as well as to each other. Both the ingredient as well as the recipe-based approaches perform similarly, while our hybrid approach outperforms the individual approaches. We show that the automatic approaches lie within an acceptable range to the CO2-values calculated manually and therefore are serious alternatives. Using the approaches suggested in this paper, the cost of calculating CO2-footprints of recipes can be reduced dramatically, which makes it possible to make this information available to a large audience. The company Eaternity, which has commercialized a CO2-calculation service based on the approaches we describe, reports that it realizes a reduction in the calculation effort of 50-60% and an overall cost reduction of 80% compared to their old, manual process.

2. Related Work

Processing and more specifically choosing, designing, adapting and comparing cooking recipes has proven popular with case-based reasoning (CBR) researchers ever since the two automated meal recommendation systems CHEF (Hammond, 1986) and Julia (Hinrichs, 1989) have been presented. Many efforts are related to the Computer Cooking Contest, which runs since 2007. We distinguish between work to automatically process the ingredients of cooking recipes and work that deals with the similarity of recipes.

Several publications deal with automatically constructing a process flow graph of a given recipe (Hamada et al., 2000), (Walter et al., 2011). Hamada et al. (2000) create domain-specific dictionaries and match the keywords in the recipe to the words in the dictionaries. Based on the structure of the sentences they then construct the process flow graph. Walter et al. (2011) preprocess and annotate the recipes with GATE, a natural language processing (NLP) framework. Based on rules created from a domain expert the ingredients, as well as the actions, are linked to a workflow. Moreover, Müller et al. (2012) automatically match the ingredients of a recipe to a nutrition database in order to estimate the nutritional value of the recipe. The similarity of recipes is mostly investigated for content-based recommender systems (Teng et al., 2012), (van Pinxteren et al., 2011).

The CO2-database that we use in our experiments as well as the whole CO2-application is described by O'Connor et al. (2018), while other CO2-reduction experiments that are conducted using the automatic ingredient matching approach are described by Itten et al. (2018).

Gonzalez and Laureano-Ortiz (1992) propose a CBR system that automatically estimates the value of a property based on similar real estates handled in past experiences. If the markets for particular properties are too sparse, they use heuristic knowledge.

The K-nearest neighbor (kNN) approach is usually applied to solve classification problems where the only prerequisite is the definition of a similarity of feature vectors. It was first mentioned in a technical report in 1951 (Fix and Hodges Jr, 1951). Since then, kNN is also used for text classification amongst others by Yang (1999) and Sebastiani (2002). In this paper, we do not classify the recipe but only use the idea of nearest neighbors in order to estimate the CO2-value based on them.

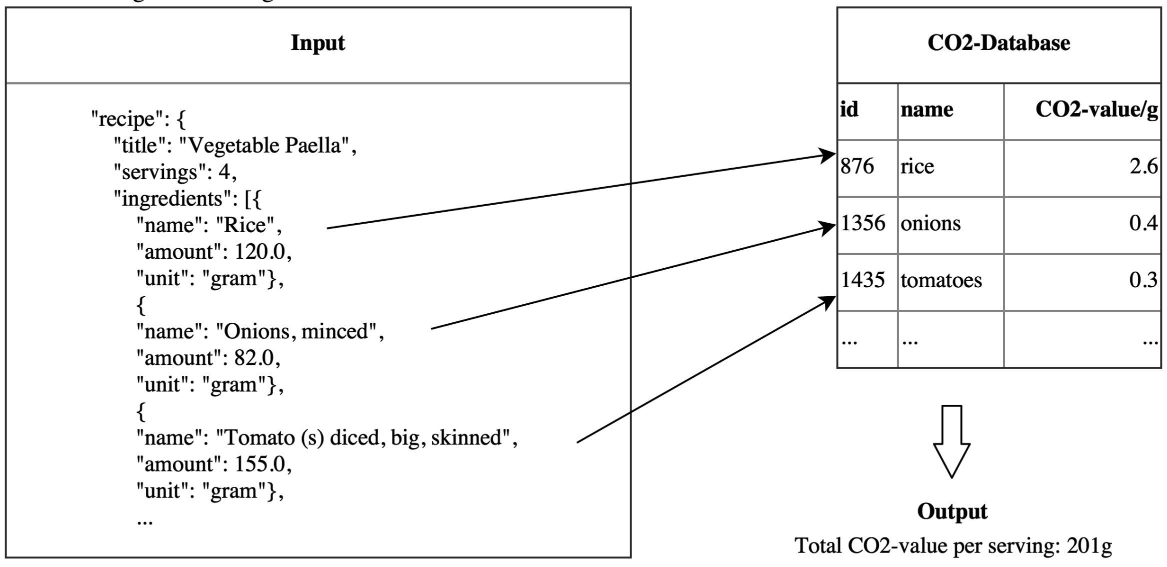
3. Methods

The case of calculating CO2-footprints is interesting for IR research on multiple fronts. As described above, in an *element-wise* approach the value is retrieved by either manually or automatically matching all the ingredient descriptions in the recipe to the appropriate food products in an LR. However, this matching is more challenging than it may appear at first glance. The difficulty stems from the fact that recipes are usually written in natural language and are therefore not restricted to use the fixed vocabulary used in the food product database. The following challenges are all very well known in NLP and IR.

The first challenge is that the authors might use synonyms in order to describe the ingredient; e.g. the ingredient description for "chard" in German might be either "Mangold" or "Krautstiel". In the cooking domain, synonyms are frequently used due to regional differences. The second challenge is the specificity of the ingredient description; both cases, very unspecific descriptions and overly specific descriptions are hard to handle correctly. For example, the unspecific description "fillet of fish" has a lot of different options to be interpreted by the cook and therefore the correct assignment to a food product in the database is a non-trivial choice. On the other hand, the description "Pinot Noir" may be too specific and in order to correctly match it to a food product in the database, the fact that this is a red wine has to be known. The third challenge is the handling of combined products, such as a pizza dough, which themselves consist of several other products.

In order to match the combined products to the food products in the database, the database either needs to contain them as well or a process to recursively split them into their base food products has to be defined. In addition to the three challenges described, we have to handle different word forms, word compounds, special characters, etc. Moreover, the food products used in recipes worldwide are manifold and it has been shown that most of them only appear in very few recipes (Müller et al., 2012). Also, new

Figure 1: Visualization of the manual as well as the automatic ingredient matching approach on an explanatory excerpt of a recipe. Note that the CO2-values are simplified for this example. In the real application, they differ depending on the season and the origin of the ingredients.



```
"recipe": {
  "title": "Vegetable Paella",
  "servings": 4,
  "ingredients": [{
    "name": "Rice",
    "amount": 120.0,
    "unit": "gram"},
    {
    "name": "Onions, minced",
    "amount": 82.0,
    "unit": "gram"},
    {
    "name": "Tomato (s) diced, big, skinned",
    "amount": 155.0,
    "unit": "gram"},
    ...
```

id	name	CO2-value/g
876	rice	2.6
1356	onions	0.4
1435	tomatoes	0.3
...	...	...

Total CO2-value per serving: 201g

products are continuously introduced to the marketplace. According to these facts, a food product database is basically never complete.

In the following sections, we propose three approaches to automatically calculate the CO2-value of cooking recipes using NLP and IR methods. Section 3.1. describes our first approach, which we call *ingredient matching*. It reproduces an automatic version of the traditional manual process of assigning CO2-values to ingredient descriptions. Section 3.2. describes the *recipe matching* approach, which estimates the CO2-value of a recipe based on similar recipes rather than the individual ingredients. Therefore, it does not depend on the completeness of the food product database. Our third approach, the *hybrid approach*, combines the ingredient and the recipe matching approaches and therefore benefits from the advantages of both. It is described in Section 3.3.

3.1. Ingredient Matching

The ingredient matching approach matches all ingredient descriptions individually to the corresponding food product in the CO2-database (Eaternity AG, 2017) and the estimate is computed by the sum of the CO2-values of each food product multiplied with the corresponding amount. As shown in Figure 1, the input is a semi-structured recipe and the output is a mapping of the ingredients to the food products in the CO2-database as well as the total CO2-value of the recipe per serving.

For each ingredient description in the German input recipes, we find the best matching food product in the database. Therefore, we create an index of food products using a traditional IR pipeline including stemming, stopword removal, decompounding and synonym handling. Apart from the commonly used stopwords, we also use several domain-specific stopwords such as pasteurized, portion and minced. To ensure a high precision for the matching, we use a light stemmer (Savoy, 2002). Since the experiments are based on German recipes, we employ a decompounding component that splits compound words such as “Zitronensaft” (lemon juice) into their components “Zitrone” (lemon) and “Saft” (juice) using a dictionary-based n-gram decompounder with a minimum word size of 10 and a minimum and maximum word constituent size of 4 respectively 12. Along with the original food products, we also index their synonyms with the same CO2-information. Moreover, we also add combined products to the index, for which the CO2-values are manually pre-calculated.

Table 1: Notation used for BM25 and our adaptations thereof.

d_j	single document
q	single query
Φ	indexing vocabulary
φ_k	single indexing feature
l_j	length of document
Δ	average document length
$\Phi(d_j)$	set of features representing document d_j
$\Phi(q)$	set of features representing query q
$w(\varphi_k, d_j)$	weight of feature φ_k for document d_j
$w(\varphi_k, q, d_j)$	weight of feature φ_k for query q and d_j
$\text{ff}(\varphi_k, d_j)$	feature frequency of feature φ_k for d_j
$\text{df}(\varphi_k)$	document frequency of feature φ_k

The search in the index is performed using an adaptation of the BM25 weighting scheme (Robertson and Zaragoza, 2009) that ignores the inverse document frequency. Unlike in most other IR applications, the fact that a term appears often in the collection does *not* mean that it is less important. For example, the database might contain several products containing the term “apple”, such as apple, apple juice, and apple puree. However, the terms juice and puree should not be weighted heavier than apple, since a match to one of the three food products containing apple is already a much better fit than a match to for example orange juice. On the other hand, the term frequency is needed since some ingredient descriptions contain the same stemmed term multiple times and thus we assume that it is indeed more important than others. Since the number of terms in the ingredient description varies, we apply a document length normalization. Hence, we use the retrieval status value (RSV) of document d_j w.r.t. query q according to BM25

$$w(\varphi_k, d_j) := \frac{\mathrm{ff}(\varphi_k, d_j)}{k_1((1-b) + b\frac{l_j}{\Delta}) + \mathrm{ff}(\varphi_k, d_j)} \cdot \log\left(\frac{0.5 + N - \mathrm{df}(\varphi_k)}{0.5 + \mathrm{df}(\varphi_k)}\right) \quad (1)$$

$$w(\varphi_k, q) := \mathrm{ff}(\varphi_k, q) \quad (2)$$

$$\mathrm{RSV}(q, d_j) := \sum_{\varphi_k \in \Phi(q) \cap \Phi(d_j)} w(\varphi_k, d_j) \cdot w(\varphi_k, q), \quad (3)$$

where we set $\mathrm{df}(\varphi_k) = 1$ for all features φ_k. Table 1 shows an overview of the notation used. Apart from the ignored inverse document frequency, we employ BM25 with the commonly used term frequency saturation parameter $k_1 = 1.2$ and document length normalization parameter $b = 0.75$. The default parameters are used due to the lack of suitable training data and to avoid overfitting. For the ingredient descriptions for which no food product can be retrieved from the index, we assign an artificial food product that has an average CO2-value.

3.2. Recipe Matching

The goal of our recipe matching approach is to estimate the CO2-footprint of an arbitrary recipe from the most similar recipe in a database of recipes for which the CO2-footprints are already known. Hence, we exploit the knowledge we already gathered with either a manual or a semi-automatic process that allocates the CO2-values. Unlike the ingredient matching, the recipe matching does not rely on assigning the individual ingredients to a database entry. Therefore, this nearest neighbor approach overcomes the long tail problem introduced by the incompleteness of the ingredient database.

An approach using IR techniques to find the most similar recipe is to run a textual search with the description of the ingredients in the query recipe against an index in which the recipes are indexed with all their ingredient descriptions. However, the similarity used in this approach does not reflect the amounts of the ingredients in the recipes, e.g. a recipe with 500g flour and 3g salt would be similar to a recipe with 500g salt and 3g flour.

Therefore, we suggest a method that also considers the amounts as an additional information, so that recipes have a higher similarity if the difference between the amounts of their respective ingredient descriptions is small. Our approach is based on an adjustment of the BM25 weighting scheme although other popular weighting schemes such as language models or divergences from randomness could be used. We adapt the weight of the query terms so that the difference between amounts of the ingredients in the query recipe and the document recipes is considered. A query term, i.e. an ingredient description, is weighted with the reciprocal difference between the amounts of the two the ingredients. Hereby, we choose the formula so that a difference of zero leads to a weight of one. Therefore, we also store the amounts of each term in each recipe, so that we can quickly retrieve the amount of a term in a given recipe when comparing recipes. In case an ingredient description consists of several terms, the amount of the ingredient will be assigned to each of its terms.

The retrieval status value (RSV) of document d_j w.r.t. query q of the adjusted BM25 that considers the amounts of the ingredients is therefore defined as:

$$w(\varphi_k, d_j) := \frac{\mathrm{ff}(\varphi_k, d_j)}{k_1((1-b) + b\frac{l_j}{\Delta}) + \mathrm{ff}(\varphi_k, d_j)} \cdot \log\left(\frac{0.5 + N - \mathrm{df}(\varphi_k)}{0.5 + \mathrm{df}(\varphi_k)}\right) \quad (4)$$

$$w(\varphi_k, q, d_j) := \frac{\mathrm{ff}(\varphi_k, q)}{|\mathrm{a}(\varphi_k, q) - \mathrm{a}(\varphi_k, d_j)| \cdot \alpha + 1} \quad (5)$$

$$\mathrm{RSV}_{\mathrm{BM25}}(q, d_j) := \sum_{\varphi_k \in \Phi(q) \cap \Phi(d_j)} w(\varphi_k, d_j) \cdot w(\varphi_k, q, d_j), \quad (6)$$

where $\mathrm{a}(\varphi_k, r)$ is the amount of the term k in the recipe r and α is a tuning parameter to weight the difference between the amounts. The *tf* saturation parameter k_1 and the document length normalization parameter b are used as in the original definition of BM25. Note, that only the definition of $w(\varphi_k, q, d_j)$ is different to the one in the original BM25, where it is equal to $\mathrm{ff}(\varphi_k, q)$.

Once the most similar recipe is known, we can use its CO2-value as an approximation of the CO2-value of the input recipe.

3.3. Hybrid Matching

The use of a hybrid approach is motivated by the failure analysis of the two individual approaches. Our goal is to obtain a more robust estimate that reduces the number of outliers, where the automatically generated value is far from the correct, manual assessment. Table 2 summarizes the reasons why the ingredient matching and recipe matching approaches produce inaccurate estimates which are outside of an acceptable range with respect to the manually computed value. The ingredient matching results in a bad estimate when one or several ingredient descriptions can not be matched to the correct food product in the database. The reason is either that the correct food product does not exist in the database (long tail problem) or that the IR pipeline fails to retrieve the correct food product. Generally, the estimates do not lie within an acceptable range either if many ingredient descriptions are not correctly matched; i.e. the error accumulates; or if a few ingredient descriptions with a high CO2-impact are matched to food products with a low CO2-impact or vice versa.

There are three main reasons for the recipe matching to produce an estimate that does not lie within an acceptable range. The first reason is that the search space in which the recipe matching approach finds the nearest neighbor often has regions in the vector space in which it is not dense. In these regions, the distance between the recipe and its nearest neighbor is bigger than in other regions where the search

Table 2: Summarized reasons for estimation errors.

Ingredient Matching	Recipe Matching
Long Tail Problem	Sparse Space Problem
IR Pipeline Problem	IR Pipeline Problem
	Granularity Problem

space is less sparse. In the recipe domain, the different regions in the vector space might also correspond to cultural differences. For example, our test collection contains a lot of Swiss menus and not so many Asian recipes; therefore, in general, the estimates for Asian menus are less accurate than for Swiss menus. The second reason for bad estimates is that the true nearest neighbor can not be retrieved since it uses a different vocabulary. The third reason for estimates that are far from the manually calculated value can be summarized as granularity problems. This means that the recipe matching, which operates on the whole recipe rather than on the individual elements, fails to produce a good estimate if the nearest neighbor recipe is similar to the input recipe, although there are small but decisive differences in the ingredients that lead to a completely different CO2-footprint. The different kinds of failures of the two approaches lead to situations where either only one of the approaches produces an estimate that is rather far from the ground truth or that one overestimates and the other underestimates the CO2-value. Therefore, we propose a hybrid matching approach that uses the average of the two CO2-estimates of the ingredient matching and the recipe matching as a new estimate, as shown in equation 7.

$$\text{estimate}_{\text{hybrid}} = \frac{\text{estimate}_{\text{ingredient}} + \text{estimate}_{\text{recipe}}}{2} \quad (7)$$

This flattens the outliers produced by the individual approaches and makes sure the system can provide a CO2-estimate for more recipes.

4. Test Collections and Language Resources

For the experiments, we use two collections of recipes that were created specifically for this task. The first collection, the so-called *hobby collection*, consists of 243 vegetarian and vegan recipes from chefkoch[1], an online platform for recipes. The second collection, the *catering collection*, contains 600 recipes from the catering company "Compass Group (Schweiz) AG", a subsidiary of Compass Group PLC, the largest caterer worldwide. The recipes in both collections are in German and are in a semi-structured form given by either chefkoch or the catering company's enterprise resource planning system. This means, each instruction line is provided with separate fields for amount, unit and ingredient description, hence no information extraction is needed. Different units, such as the number of teaspoons, are converted to grams using a simple set of rules.

The ingredient matching approach matches the ingredient descriptions to a product index that contains 3,121 food products that was generated from the LR (Eaternity AG, 2017). The LR contains base food products with their CO2-values as well as synonyms that are linked to the base food products. The LR contains a lot of very region specific food products, such as "Cervelat" and "Roesti" which are frequently used in cooking recipes in Switzerland. The recipe matching approach searches for the nearest neighbor recipes in a recipe index that contains approximately 50,000 recipes. Most of the recipes are from catering companies others are from chefkoch and various other sources. Both the product index and the recipe index are primarily in German.

We manually built a ground truth for both the hobby and the catering collections. That means that for each ingredient in each recipe we manually assigned the best matching food product in the product index. Based on this ground truth it is possible to calculate the CO2-footprint of each recipe in the collection. For example, "spaghetti carbonara" has an expected CO2-value of 774g. There are also recipes with a much larger CO2-value such as "schnitzel with french fries" which has a CO2-value of 2,366g. Table 3 shows the range of the CO2-values of the recipes in these collections. The hobby collection has a significantly smaller average CO2-value per recipe (1,100g) than the catering collection (1,700g) since the hobby collection only contains vegan and vegetarian recipes.

Table 3: Statistics of the test collections.

Collection	Catering	Hobby
Number of recipes	600	243
Minimum CO2-value	32g	113g
Maximum CO2-value	13,513g	1,732g
Average CO2-value	1,700g	1,100g

5. Experiments

5.1. Ingredient Matching

The ingredient matching approach matches the ingredient descriptions to the food products in the database. Table 4 shows the precision, the fraction of correctly matched ingredient descriptions, as well as the mean absolute error (MAE) and the Pearson correlation between the CO2-value estimate from the ingredient matching and the CO2-values from the manual matching. Hereby, correctly matched means that the automatic matching is strictly equal to the manual matching. The mean absolute error is the average of all the absolute errors in the test collection, where the absolute error of a recipe is the difference between the expected CO2-value and our estimate. For example, "spaghetti carbonara" has an expected CO2-value of 774g and an estimate of 684g which results in an absolute error of 90g. The Pearson correlation measures the linear dependence between two variables, in our case the manually assessed CO2-values and the estimates from the automatic process. The possible values are between 1 and -1, where 1 is the maximal positive correlation, 0 means no correlation and -1 is the maximal negative correlation.

The precision for the catering collection is slightly higher than for the hobby collection, mostly since the food product database was mainly designed for catering recipes. At

[1]http://www.chefkoch.de/

Table 4: Matching results using the ingredient matching approach on the two test collections *catering* and *hobby*.

Collection	Catering	Hobby
Precision	0.72	0.68
Mean absolute error	336g	163g
Pearson correlation	0.81	0.73

first glance, the achieved precisions of 0.72 respectively 0.68 are not that encouraging. However, given that other studies show that even the consensus of human assessors is smaller than 75% for 23% of the recipes (Müller et al., 2012), the achieved precision is at least acceptable. Having a closer look at some of the wrongly matched ingredients descriptions, we indeed find many examples which are within the margin of human disagreement. For example, “celery large” is wrongly matched to “celery root” instead of “celery stalks” as denoted in the ground truth, although both seem to be valid options. There are however also some IR specific issues. For example, “red trout fillet (breed)” is wrongly matched to “salmon trout (breed, fillet)” rather than “trout”.
The significantly smaller average CO2-value per recipe in the hobby collection, as shown in Table 3, is the main reason why the MAE of the hobby collection is smaller than the MAE of the catering collection.

5.2. Recipe Matching

The recipe matching approach, in which we estimate the CO2-value of an input recipe by its most similar recipe, heavily relies on the size of the recipe corpus from which the similar recipes are retrieved. Our retrieval system is built on top of Lucene and is using the built-in BM25 weighting scheme with the default saturation parameter $k_1 = 1.2$ and the document length normalization parameter $b = 0.75$.
Table 5 shows the MAE and the correlation between the CO2-value estimate from the recipe matching and the CO2-values from the manual matching. For the experiments, we use $\alpha = 0.02$ as the tuning parameter of the weight of the difference between the amounts. Note that our ground truth does not include the closest neighbor of the recipes, but only the manually assigned food products and the total CO2-value of each recipe, thus we do not specify the precision for the recipe matching approach.

Table 5: Matching results using the recipe matching approach on the two test collections *catering* and *hobby*.

Collection	Catering	Hobby
Mean absolute error	310g	360g
Pearson correlation	0.83	0.14

In order to explain the different performances of the algorithm on the two datasets, we first have a look at the two collections. As already stated previously the hobby collection only contains vegan and vegetarian recipes from a hobby cooking platform, while the catering collection contains recipes from several canteens in Switzerland. Having a closer look shows that the two collections are quite different regarding the number of ingredients used in each recipe. An average recipe in the hobby collection consists of 12.7 ingredients and an average recipe in the catering collection has 20.5 ingredients. However, not only the number of ingredients is different but also the ingredients themselves. Therefore the most similar recipe from which the CO2-value is used as an estimate is most likely a recipe from the same category (hobby or catering) as the input recipe. The corpus used to retrieve the recipes with already allocated CO2-values consists of approximately 50,000 recipes from which only around 1% are recipes from the hobby domain, while all the others stem from the catering domain. The lack of close neighbors; i.e. too few recipes from the hobby domain, therefore explains the small correlation (0.14) of estimates in the hobby collection. Even though the performance of the recipe matching for the hobby collection is not as good as for the catering collection, the MAE for the hobby collection (360g) is still in the same range as for the catering collection (310g) due to the smaller average CO2-value of the vegan and vegetarian hobby recipes.

5.3. Hybrid Matching

The hybrid matching approach combines the ingredient matching and recipe matching by averaging the two estimates and therefore is able to account for their individual shortcomings. Table 6 shows the MAE and the correlation between the CO2-value estimate from the hybrid matching and the CO2-values from the manual matching.

Table 6: Matching results using the three matching approaches on the two test collections *catering* and *hobby*.

Method	Measure	Catering	Hobby
Ingredient	Precision	0.72	0.68
	Mean absolute error	336g	163g
	Pearson correlation	0.81	0.73
Recipe	Mean absolute error	310g	360g
	Pearson correlation	0.83	0.14
Hybrid	Mean absolute error	279g	206g
	Pearson correlation	0.90	0.55

For the catering collection the hybrid matching approach achieves a better result for both measures (MAE and correlation) than the other approaches individually. In spite of the significantly worse performance of the recipe matching for the hobby collection the hybrid matching only achieves a slightly worse result as the ingredient matching. These results show that in the case in which both individual approaches achieve an acceptable performance the hybrid matching results in more reliable estimates.

6. Conclusions

We proposed three approaches using IR to automatically compute a single numerical value of a semi-structured item that consists of a list of textual elements based on the use case of calculating CO2-footprints of cooking recipes,
The first approach, ingredient matching, calculates the CO2-footprint on an element-basis; i.e. the ingredients.

Our experiments show that the CO2-value estimates of the ingredient matching lie within an acceptable range compared to the estimate of the manual calculation. The second approach estimates the CO2-values based on similar recipes rather than individual ingredients. Since the estimate is no longer based on the individual ingredients, this recipe matching approach overcomes the long tail problem of the ingredient matching, i.e. that the food product LR is most likely not complete.

For the similarity of recipes, we proposed an adaptation of the BM25 weighting scheme that takes the different amounts of the ingredients into account. We showed that the recipe matching slightly outperforms the ingredient matching, if the recipe corpus is large enough. We have reason to believe, that the effectiveness of matching would increase as the size of the collection of recipes that is searched against increases.

Combining both the ingredient matching and the recipe matching with our hybrid approach allows us to estimate the CO2-value even more accurately. It is therefore able to balance the shortcomings of the individual approaches. The achieved correlation of 0.9 between the CO2-value estimates of the hybrid matching and the CO2-value estimates of the manual matching shows that the automatic calculation is a serious alternative to the manual calculation and therefore the costs of a manual calculation can be reduced dramatically by instantiating the automatic calculation. Indeed, first experiences from using the approaches in the commercial CO2-calculation service of our partner Eaternity indicate a reduction in effort for the calculations in the range of 50-60%, with an even higher overall cost reduction of 80%.

As a next step, the accuracy of the estimates of the hybrid matching approach could possibly be further improved by weighting the estimates of the ingredient and the recipe matching based on an estimate of their reliability. The reliability of the CO2-estimates of the recipe matching could for example be predicted using the distance between the input recipe and its nearest neighbor.

7. Acknowledgments

We thank Compass Group (Schweiz) for the recipe dataset and Eaternity AG for the project work in which most of the approaches presented in this paper were developed.

8. Bibliographical References

Fix, E. and Hodges Jr, J. L. (1951). Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, DTIC Document.

Gonzalez, A. J. and Laureano-Ortiz, R. (1992). A case-based reasoning approach to real estate property appraisal. *Expert Systems with Applications*, 4(2):229 – 246.

Hamada, R., Ide, I., Sakai, S., and Tanaka, H. (2000). Structural analysis of cooking preparation steps in japanese. In *Proceedings of the fifth international workshop on on Information retrieval with Asian languages*, pages 157–164. ACM.

Hammond, K. J. (1986). Chef: A model of case-based planning. In *AAAI*, pages 267–271.

Hinrichs, T. R. (1989). Strategies for adaptation and recovery in a design problem solver. In *Proc. of the 2 nd Workshop on Case-Based Reasoning*, pages 115–118.

Itten, R., Imhof, M., Jaggi, A., and Stucki, M. (2018). *Combining Natural Language Processing and Life Cycle Assessment for Computer-Aided Optimisation of Greenhouse Gas Emissions in System Catering*. unpublished manuscript.

Müller, M., Harvey, M., Elsweiler, D., and Mika, S. (2012). Ingredient matching to determine the nutritional properties of internet-sourced recipes. In *2012 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, pages 73–80. IEEE.

O'Connor, I., Aleksandrowicz, A., Scheuss, A., Jaggi, A., Klarmann, M., and Ellens, J. (2018). *Description of the Eaternity Database*. unpublished manuscript.

Robertson, S. and Zaragoza, H. (2009). *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Savoy, J. (2002). *Morphologie et recherche d'information*. Université de Neuchâtel, Faculté de droit et des sciences économiques, Division économique et sociale.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

Teng, C.-Y., Lin, Y.-R., and Adamic, L. A. (2012). Recipe recommendation using ingredient networks. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 298–307. ACM.

Tukker, A. and Jansen, B. (2006). Environmental impacts of products: A detailed review of studies. *Journal of Industrial Ecology*, 10(3):159–182.

van Pinxteren, Y., Geleijnse, G., and Kamsteeg, P. (2011). Deriving a recipe similarity measure for recommending healthful meals. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 105–114. ACM.

Walter, K., Minor, M., and Bergmann, R. (2011). Workflow extraction from cooking recipes. In *Proceedings of the ICCBR 2011 Workshops*, pages 207–216.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90.

Zipf, G. K. (1949). Human behavior and the principle of least effort.

9. Language Resource References

Eaternity AG. (2017). *Eaternity Food Product Database*. Eaternity AG, not publicly available, 1.0.