pic2kcal: End-to-End Calorie Estimation From Pictures of Food

Robin Ruede, Lukas Frank, Verena Heußer
Karlsruhe Institute of Technology

Abstract

Latest approaches to predict calories of food are using mostly models which are composed out a several pipeline steps like segment the image, estimate the weight and classify the ingredient. We present in this paper a novel end-to-end approach to estimate the kcal directly from a picture. Since there is no large scale public available dataset to train models on this task we furthermore collected data of recipes including images and matched the ingredients of the recipes with ground truth nutritional information of a food database.

1. Introduction

The last years the awareness of healthier and more balanced diets has risen a lot. For the user it is often hard to keep track of the consumed calories or related with a lot of manually work to find recipes with calorie information or to look up the nutritional information of each ingredient. Even commercial offered apps often needs supervision of the user to select the recognized components of the meal.

Datasets play an important role to solve computer vision problems. It is crucial to have enough trainings data for the very deep CNN models with a lot parameters. In this work we present an entirely new food dataset which could be used as training or benchmark for machine learning models. The dataset contains roughly xxxx recipes including nutritional information as well as ingredients list and xxx images of prepared meals.

We used the dataset to show that the mapping from food image to kcal information could be learned by a neural network. Experiments have shown that injecting further information into the model in form of top n ingredients outperform the models with less information.

2. Related Work

There's some other papers like [1; 2;]. Ours is more end to end and also BETTER

- [1]:
  - details von verena, wsl zu detailliert:
  - supervised, pipeline: predict food-type -> predict size in g -> predict kcal based on size and food-type (nicht end-to-end)
  - Dataset: Pittsburg fast-food image dataset (61Essens-Katergorien, 101Sub-Kategorien, jew. 3 Instanzen des Gerichts -> Annotiert mit Größe und Art des Gerichts; hier nur Subset verwendet (6 Klassen angerichten, ˜1100 Bilder))
  - Architektur / Ablauf
    - PCA: auf 23Features runter
    - Food-type-Classifier: verschiedene Varianten ausprobiert; beste: SVM
    - Size Predictor: beste: Random forests
    - Calories Predictor: NN als predictor
      - input: 23 visual features + food type + food size
      - Performance: Vergleich nur gegen eine eigene Studie, keine Angabe von Accuracy (Vergleich des pipeline-Ansatzes (visualfeatures+type+size) gegen Modell nur trainiert auf vis. features
    - Handgemachte Annotationen für die Kcal-Angaben
- [3]:
  - wsl zu detailliert
  - Restaurant specific im2calories
  - Task1: Is food in Image ? -> Food vs. non-food (binäre Klassifikation)
    - mit googLeNet CNN (pretrained on ImageNet)
  - Task2: Content Analysis -> Calorie Prediction
    - Training:
      1. vorhersage der Zutaten über multi-label classifier
      2. Lookup der Zutaten für mapping Zutat -> kcal
      3. dann schätzen der Gesamt-kcal-anzahl (Summe über die Zutaten-kcal)
    - Test: auf Datensatz 'MenuMatch'

  - Mapping Zutat -> Kcal über FNDDS

- calorie mama: recognizes ingredients and meals from pictures. pretty impressive tbh.
- [4]: "in which they searched the calorie-annotatedfood photo database for the top 5 similar images based onconventional hand-crafted features such as SURF-based BoFand color histograms and estimated food calories by averag-ing the food calories of the top 5 food photos"
- [5]: recipe generation (ingredients list, instructions, no amounts or kcal)
- [2]: multi-task VGG: kcal estimation, food categorization, ingredients estimation, cooking instructions. probably closest to ours? ingredients are not predicted individually, but as a single averaged word2vec embedding to make kcal prediction better
- [6]: comparison of many different things: used datasets, segmentation methods, classification approaches, volume estimation methods of 10+ other papers

3. Dataset Extraction and Preprocessing

3.1. Collection

We collected a dataset from the a popular German recipe website that contains ingredient lists, cooking instructions, and pictures of the resulting meals. The recipes are from many different cuisines and also include things like cakes, cocktails, and other. Most recipes have at least one picture. Pictures can come both from the original author of the recipe, as well as from third parties. Some of the pictures are of a single plate of food, others are for example of a whole casserole. We do not have any information about whether a picture contains a single portion. Around 10% of recipes contain a user-given value for how many calories per portion the recipe has.

3.2. Matching / Preprocessing

Since the dataset only has user-given calorie information for a small part of the data and doesn't include any details regarding the macronutrient composition, and since the user given information is often inaccurate [todo: compare to ours??], we decided to match the list of ingredients against a database of nutritional values to sum up the proportions of macronutrients as well as the total calories.

To facilitate this, we collected a secondary dataset from a German website of nutritional values. The website contains values for the amount of fat, protein, and carbohydrates in grams per 100g of product. Additionally, it contains user-sourced food amounts like "1 medium-sized apple = 130g". The data is partially sourced from the USDA Food Composition Database [7], and partially crowd sourced from manufacturer-given data of specific products.

Matching the recipe ingredients to the nutritional database has two main problems.

Firstly, the given ingredient name often includes information that is not relevant to the product itself, but rather to its preparation or visual qualities. These additional text snippets are hard to separate from information that is relevant. For example: `3 onions, diced` and `3 onions, in slices` refer to the same product, while `500g pasta, cooked` and `500g pasta, raw` vary significantly in their caloric density. We experimented with three approaches [reformulate] to solve this problem. Firstly, we tried simply matching the ingredient to the nearest ingredient based on character edit distance (Levenshtein distance). This resulted in very bad matchings because of missing handling of synonyms and the above issue. To solve this we tokenize the ingredient name to words, embed each word to a vector with Word2Vec or FastText [8], and then use average the word vectors to get a ingredient vector. This is the same method as used in the fasttext library for extracting sentence vectors. This still lead to unsatisfactory results, since each word in the ingredient name has the same weight, even though some specify less important details. For example in "red onion" vs "red apple", the word "red" is much less important than "onion" and "apple". We got the best result by using the Google Universal Sentence Encoder [9], which creates 512-dimensional embeddings of any amount of text. We find the best matches for a ingredient by comparing the embedding of user-given free text from the recipe to the embeddings for all food items for which we have nutritional data using the cosine distance, and then try to find a conversion for the given amount to a normalized gram or milliliter amount.

The second problem is matching the amounts. For ingredients given in grams this is trivial, but for many items the recipe authors use other units of measure like e.g. can, piece, tablespoon, "some", "2 large X", "salt 'by taste' ". Since spices usually have little impact on the nutritional values, we exclude ingredients that are "by taste" and similar. For the other amounts, we match the unit name (like tablespoon or "medium large") exactly and multiply it with the given amount. We also add some special cases like matching "can" to "can (drained weight)" and similar.

The amount matching is applied to all possible ingredient matches that are similar by more than 84% (measured by cosine distance) [why lol] in decending order, or to the single closest ingredient if there is no match more accurate than 84%.

If the amount matching fails, the ingredient is marked as unmatched. If a recipe has at least one unmatched ingredient, it is discarded.

3.3. Dataset Statistics

In total, the recipe website contains 330 thousand recipes. Of these, 210 thousand have at least one picture. Around 20 thousand recipes with pictures have user-given calorie information, though we didn't use these in the end. The recipes contain a total of 374 thousand unique ingredients. This high number is caused by slight differences in spelling or irrelevant details. In total, we collected 900 thousand pictures. On average, each recipe has 3 pictures.

The database of nutritional values contains a total of 390 thousand ingredients. Many of these are incomplete or duplicates, so we filter them by popularity to 123 thousand ingredients.

After matching the ingredients to the recipes, we have 85 thousand recipes with full nutritional information. We lose 60% of recipes during matching because our matching discards recipes quickly when the ingredients don't fully match. This is so we can ensure we only retain data points that are accurate. This could be improved with further tweaking.

In total, we have 270 thousand data points (because each recipe has multiple images). We split these into train, validation and test set so that multiple pictures of the same recipe are in the same data split.

The 20 most common ingredients are shown in Table 1. Note how common baking ingredients are. This indicates a cake bias, i.e. the dataset may be biased towards sweet meals and desserts.

Count	Ingredient
119244	Salz
59066	Zucker
58185	Ei, vom Huhn
46069	Mehl
45891	Butter
41206	Zwiebel, frisch
24531	Milch (3,8 %)
24011	Vanillezucker
23476	Zucker
22822	Öl
22781	Paprika, orange
21348	Knoblauch
20359	Wasser
19935	Knoblauch, frisch
19336	Pfefferbreze
18928	Olivenöl
15966	Backpulver
15039	Sahne
14751	Zitrone, frisch
13077	Paprikapulver
12487	Gemüsebrühe, pflanzlich
12136	Backpulver
11960	Käse
11673	Kartoffeln
10926	Eigelb, vom Huhn
10780	Butter, Durchschnittswert
9591	Puderzucker
9439	Petersilie, frisch
8708	Zucchini, grün, frisch
8630	Öl
8293	Mehl, Weizenmehl Typ 405

Table 1. Most common ingredients after matching.

4. Models

We followed an end-to-end approach to solve the calorie prediction problem of food images. To do so we used a pretrained ResNet and DenseNet architecture. We kept the feature extractor layers and replaced the last fully-connected classification layer. We try to solve the problem interpreting it on the one hand as a classification task and on the other hand as a regression problem. Furthermore we introduced additional learning feedback following a multi-task approach.

We describe in the following only the last layer of the neural network.

In the regression case we trained a model predicting only the kcal information with one output neuron and another to predict additionally protein, fat and carbohydrates information using four neurons. The two models were trained using a L1 and smooth L1 loss.

We transformed the two already described models to a classification problem quantizing the regression outputs. So we introduced 50 class buckets for each regression output. The models were trained using a cross entropy loss.

The multi-task model is based on the regression model including the nutritional information with additional binary outputs to predict the top n ingredients. The resulting layer has four regression outputs with 50 binary outputs. The used loss combines a smooth L1 loss for the regression outputs and an binary cross entropy for the top 50 ingredients. To get the same scaling of the two learning signals we scaled the binary cross entropy loss with a factor of 400.

$$loss = L1 + 400 * BCE$$

There are no reference papers with a similar task like we do therefore we could not compare our results to other implementations. Due to we implemented a simple baseline to get an evidence that our model actually learns something and is better than guessing.

The baseline for the kcal prediction basically is the mean of all samples in the train dataset. That means that the model predicts in the inference always just the mean of the already seen kcal values. We used the same baseline for predicting the nutritional data.

5. Experiments

We divided the generated dataset into train/test/validation (xx/xx/xx) splits. Our training set contains xxx samples with around xxx images each recipe. The network was trained 40 epochs using a batch size of 50 samples each batch. The samples of the batches were shuffled every epoch and we evaluated the performance of the model every fiftieth batch. We implemented all networks using Pytorch.

To evaluate the performance of the model we trained several networks and run several experiments to evaluated them using firstly the evaluation data set to get quick feedback. To measure the performance of the model we only compared the given kcal information with the prediction of the network.

Firstly we used our raw data set to train the kcal-model. We wanted the network to predict the kcal information of the recipe visualized on the given input image. To perform well in this task the model needs to learn the concept of the recipe size and predict the calories according it. We assumed due to the amount of samples and the capacity of the model the problem is well learnable. Unfortunately the trained regression model performed not well on the task probably because of outlier recipes in our dataset with not valid kcal information provided by the users. Even after outlier removal, prediction of normalized kcal information of portion and trying a classification approach the model was only slightly better than a baseline model.

Second we evaluated if the additional nutritional information supports the networks capability to generalize on the recipe and portion size. Both the classification and regression objectives performed not well with the further information.

Lastly we reformulated the training objective to a slightly easier problem. We trained the network to predict the calory density of the visualized image. Because of the normalization the network only needs to grasp how many calories are in for instance 100g of the meal. This modification led to significant better results.

We could furthermore improve the results of the model using the multi-task approach. The top 50 ingredients of the recipes were injected as further information to support the model predicting the kcal information. We report the results of this model in the result section.

6. Results

Our results can be seen in Table 2. Example outputs can be seen in Figure 1.

Method	kcal relative error
baseline	0.464
ours (kcal only)	0.361
ours (w/ macros)	0.352
ours (w/ macros+ings)	0.328

Table 2. Results per 100g. Note that multitask learning improves performance.

Figure 1. Some example results, showing predicted calories, fat, protein, carbohydrates and ingredients.

7. Problems/Fails

8. Future Work

The dataset we extracted contains at the moment much more attributes as we are using for now. Once the dataset gets further processed the recipe instructions, the type of the meal (cake, side dish), the rating and further properties could be used for training further models. The current dataset contains at the moment all available photos of each recipe. It may make sense to implement sanity checks to filter images out if they do not match the recipe in a proper way.

Further problems related to food could be approached using the dataset. For some people it may be interesting to know if the meal contains a special ingredient because of allergies or if it is vegan or vegetarian. The dataset provides needed information to train a variety of different models to solve problems related to food.

Currently our kcal prediction model is not highly optimized for the task since it is build on top of pretrained models. As already evaluated it is benefital to inject further data into da model therefore it may be interesting to do further investigation an different model architectures. For instance the representation of the top-n ingredient neurons could be changed from a binary value to the objective predicting the volume of the ingredient. It may also make sense to build entire new architectures using kernels with a size which match the requirenments of predicting/classifying food images.

References

[1] M. Chokr and S. Elbassuoni, "Calories Prediction from Food Images," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 4664–4669.

[2] T. Ege and K. Yanai, "Image-Based Food Calorie Estimation Using Knowledge on Food Categories, Ingredients and Cooking Directions," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, 2017, pp. 367–375.

[3] A. Myers *et al.*, "Im2Calories: Towards an Automated Mobile Vision Food Diary," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1233–1241.

[4] T. Miyazaki, G. C. de Silva, and K. Aizawa, "Image-based Calorie Content Estimation for Dietary Assessment," in *2011 IEEE International Symposium on Multimedia*, 2011, pp. 363–368.

[5] A. Romero, X. Giro-i-Nieto, M. Drozdzal, and A. Salvador, "Inverse Cooking: Recipe Generation from Food Images," Dec. 2018.

[6] M. A. Subhi, S. H. Ali, and M. A. Mohammed, "Vision-Based Approaches for Automatic Food Recognition and Dietary Assessment: A Survey," *IEEE Access*, vol. 7, pp. 35370–35381, 2019.

[7] *USDA Food Composition Databases*..

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, Dec. 2017.

[9] D. Cer *et al.*, "Universal Sentence Encoder," in *In submission to: EMNLP demonstration*, 2018.