KitcheNette: Predicting and Ranking Food Ingredient Pairings using Siamese Neural Networks

Donghyeon Park, Keonwoo Kim, Yonggyu Park, Jungwoon Shin and Jaewoo Kang*

Korea University

{parkdh, akim, yongqyu, jungwoonshin, kangj}@korea.ac.kr

Abstract

As a vast number of ingredients exist in the culinary world, there are countless food ingredient pairings, but only a small number of pairings have been adopted by chefs and studied by food researchers. In this work, we propose KitcheNette which is a model that predicts food ingredient pairing scores and recommends optimal ingredient pairings. KitcheNette employs Siamese neural networks and is trained on our annotated dataset containing 300K scores of pairings generated from numerous ingredients in food recipes. As the results demonstrate, our model not only outperforms other baseline models, but also can recommend complementary food pairings and discover novel ingredient pairings.

1 Introduction

Many chefs, gourmets, and food-related researchers have focused on studying food pairing for decades. There are books [Page and Dornenburg, 2008; Dornenburg and Page, 2009] featuring a number of food pairings recommended based on accumulated experiences of professional chefs and food gourmets in the culinary world. Since food pairings are made based on the experiences of experts, food pairing itself is subjective and difficult to quantify. In the academic field, some food-related researchers [Ahn et al., 2011; Ahn and Ahnert, 2013; Garg et al., 2017; Simas et al., 2017] focused on determining the qualities of complementary food pairings based on analysis of sharing flavor compounds. However, FlavorDB built by [Garg et al., 2017] contains only a limited number of flavor compounds and natural ingredients and a considerable amount of time and effort is required to analyze the flavor compounds of food ingredient.

In this work, we introduce KitcheNette which is a model based on Siamese neural networks [Koch et al., 2015]. As shown in Figure 1, KitcheNette first trains on our annotated dataset containing more than 300k scores of known pairings, which constitute only 5% of the total possible number of pairings in our dataset. These quantified scores indicate whether each food pair is complementary or not. Then our trained model predicts the scores of unknown pairings consisting of food ingredients that have infrequently or never been used in recipes, which constitute 95% of the total number of pairings. The three unknown pairings in Figure 1 that our model found are known to be culture specific pairings in Nordic, Japanese, and Mexican cuisine, respectively. To train our model, we constructed our own dataset which contains the scores of 300k food ingredient pairings obtained from 1M human-generated cooking recipes [Salvador et al., 2017; Marin et al., 2018]. Here, the amounts and personal preferences of ingredients and the preparation process in cooking were not considered. Our model employs Siamese neural networks with wide&deep architecture designed to learn the relationship of a food pairing. We then conducted experiments to compare it with several baseline models and confirmed that our model KitcheNette outperformed all the other models.

To further evaluate our model’s prediction performance, three qualitative analyses were conducted. First, we analyzed some example cases of food pairings to test whether our model successfully predicts the scores of unknown pairings. Second, we compared the ranking results of commonly used food ingredient pairings recommended by KitcheNette with those in FlavorDB [Garg et al., 2017]. Our ranking results are more reliable and consistent with human food-pairing knowledge, compared with the results of FlavorDB. Third, we compared the food pairing recommenda-
tions of our model, which were based on predicted scores, with those of cooking experts [Page and Dornenburg, 2008; Dornenburg and Page, 2009]. We found that most of the recommendations of our model were the same as those of the cooking experts, which demonstrates the accuracy of our model. In addition, our model recommended food pairings with ingredients not commonly featured in recipes. To this extent, our work attempts to broaden the underlying concept of food pairing and introduce a data-driven, deep learning based approach for discovering novel ingredient pairs.

The major contributions are summarized as follows.

• We propose a data-driven task which involves discovering new complementary food pairings that could be used in future recipes.

• We construct a large scale dataset that contains the scores of food ingredient pairings.

• We show that KitcheNette\(^1\) which uses Siamese neural networks and wide&deep learning achieves high performance in predicting the scores of food ingredient pairings.

• Through qualitative analyses, we demonstrate KitcheNette’s effectiveness in recommending complementary food pairings\(^2\) and discovering new food pairings.

2 Related Work

2.1 Food Related Research

Researches on Discovering Food Pairings. [Ahnh et al., 2011] and [Ahn and Ahnert, 2013] introduce a flavor network where the network’s edge is built based on the number of flavor compounds shared by culinary ingredients. The flavor network is comprised of 381 ingredients and 1,021 flavor compounds. FlavorDB [Garg et al., 2017] combines existing food repositories to provide a larger database with the user-interactive page. Food-bridging [Simas et al., 2017] improves the flavor network [Ahnh et al., 2011] by adding additional bridges between two ingredients through a chain of pairwise affinities despite the chemical compound similarity of the two ingredients being low. However, they cover only a limited number of flavor compounds and natural ingredients and some well-known food pairings (e.g., red wine and beef) have very few flavor compounds in common. Our work employs a data-driven model for predicting the scores of food pairings and finding new food pairings on a larger scale.

Researches on Recommending Recipes. The recipe recommendation [Teng et al., 2012] has been proposed to determine whether a food ingredient is essential in a recipe. recipe recommendation uses two different recipe networks that can accurately predict recipe ratings. Also, finding a surprising and plausible ingredient combination is not new. [Grace et al., 2016; Grace and Maher, 2016] combines cased-based reasoning and deep learning to generate new recipe designs. Our model KitcheNette is similar in that it recommends a new combination of food ingredients, but KitcheNette trains on datasets of pairing scores on a larger scale and can suggest novel food pairings.

2.2 Siamese Neural Networks

Siamese neural networks [Koch et al., 2015] have been employed in various tasks to learn similarities between two different inputs. Also, some variations of Siamese neural networks have been introduced.

[Mueller and Thyagarajan, 2016] proposed the Manhattan LSTM model which takes two sentences as input. The Manhattan LSTM model generates vector representations for each sentence and calculates the similarity between the vector representations using the simple function \( \exp(-||h1 - h2||) \) where \( h1 \) and \( h2 \) are the embedding vectors. [Yuan et al., 2018] proposed a customized contrastive loss function that can be divided into a partial loss function for positive pairs and a partial loss function for negative pairs. The loss function widens the distance between two vector representations of negative pairs while narrowing the distance between two vector representations of positive pairs. While these two works ([Mueller and Thyagarajan, 2016; Yuan et al., 2018]) use Siamese neural networks and take two different inputs of the same type, [Liang et al., 2018]’s proposed Siamese-based model takes one input as the standard by which the other input is evaluated. The input is evaluated based on its similarity to the input used as the standard. Our model is designed to train semantic relationships of a food pair beyond simple distance-based similarity mentioned above.

3 Dataset

3.1 Dataset Description and Preprocessing

In this work, we utilized Recipe1M [Marin et al., 2018], a dataset containing approximately one million recipes and their corresponding images which were collected from multiple popular websites related to cooking. All content in Recipe1M can be divided into two categories: texts and images. The recipe texts of Recipe1M [Marin et al., 2018] consist of the following two parts: the list of ingredients and the instructions of a recipe. The Im2Recipe [Salvador et al., 2017] used a bi-directional LSTM based ingredient name extraction module that performs logistic regression on each word in the list of ingredients in Recipe1M to extract the ingredient names only apart. For instance, “2 tbsp of olive oil” is extracted as olive_oil. From the instructions for a recipe, [Marin et al., 2018] trains all the vocabulary including ingredient names with the word2vec [Mikolov et al., 2013] fashion. Among a total of 30,167 learned vocabulary, we obtained 3,567 unique ingredient names as shown in Table 1.

3.2 Food Ingredient Pairing Score Generation

Food Ingredient Pairing Score Dataset. As mentioned in the earlier sections, traditional ingredient pairing methods rely on the extensive experience and knowledge of human experts in the culinary industry. As a result, the amount of data available for training deep-learning models is insufficient. To address this problem, we construct a new large-scale dataset of food pairing scores on which deep-learning models can be trained. Food pairings are given a score from -1 to 1 based on

\(^1\)https://github.com/dmis-lab/KitcheNette

\(^2\)A demo version of user-interactive KicheNette is available at
http://kitchenette.korea.ac.kr.
Table 1: Statistics of Ingredients and Pairings. Known Pairs\(^4\) consist of ingredients whose occurrence counts are greater than 20. In addition, each known pair has a co-occurrence count of at least 5.

### 4.3 Predicting Food Ingredient Pairing Scores

In the ‘Pairing Score Prediction Component’, we employ wide\&deep learning [Cheng et al., 2016]. The layer is divided into a wide layer and a deep layer. In the deep layer, two \(j\)-dimensional learned representation vectors are concatenated and passed to another MLP that computes a joint representation of two ingredients. This joint representation is denoted as deep vector \(d\) and is mathematically expressed as follows:

\[
d = f(W_4f(W_3(h_a; h_b) + b_3) + b_4)
\]

where:

\[
h(x, y) = \log \frac{p(x, y)}{p(x)p(y)}
\]

\[
pmi(x; y) = \log \frac{p(x, y)}{p(x)p(y)}
\]

\[
h(x, y) = -\log p(x, y)
\]

### Results

As a result in Table 1, we obtained a total of 356,451 valid known pairings. All the other pairings were considered as unknown pairings. The final distribution of food ingredient pairing scores follows the approximated normal distribution. We assume that the ingredient pairing scores (0.274 or higher) in the upper 5% \((\mu + 2\sigma)\) of the distribution are the complement pairings, whereas scores with negative numbers are examples of bad pairings as shown in Table 2.

### 4 Model

#### 4.1 Learning Ingredient Representations

We propose a model that predicts the scores of ingredient pairings. Our model architecture consists of two major components, as shown in Figure 2. The first is the ‘Ingredient Representation Component’ which uses Siamese neural networks [Koch et al., 2015] where two identical multi-layer perceptrons (MLPs) with the same weights each receive a different 300-dimensional word vector representation. Each MLP has two fully connected layers which process the input ingredient vector. Let \((X_a, X_b)\) be a pair of ingredients represented as 300-dimensional word vectors. \(W_a\) and \(b_a\) are the shared weights and bias of an MLP, respectively, and \(f(\cdot)\) denotes the activation function for non-linearity. We use \(f(\cdot)(\max(\cdot, 0))\) as rectified linear units (ReLUs). The learned representations \((h_a, h_b)\) of this pair are mathematically expressed as follows:

\[
h_a = f(W_2f(W_1X_a + b_1) + b_2)
\]

\[
h_b = f(W_2f(W_1X_b + b_1) + b_2)
\]

where \(W_1 \in \mathbb{R}^{1 \times 300}, W_2 \in \mathbb{R}^{j \times i}, b_1 \in \mathbb{R}^i, \) and \(b_2 \in \mathbb{R}^j, \) and \(i\) and \(j\) are the number of hidden units.
where \( W_3 \in \mathbb{R}^{j \times 2j} \), \( W_4 \in \mathbb{R}^{j \times j} \), \( b_3 \in \mathbb{R}^j \), \( b_4 \in \mathbb{R}^j \), and \( j \) is the number of hidden units in each layer. In the \textit{wide layer}, the outer product of two \( j \)-dimensional learned representation vectors is computed and a \( j \times j \) matrix is generated. The matrix is then flattened to a \( j^2 \)-dimensional \textit{wide vector} \( w \) which is mathematically expressed as follows:

\[
w = g(h_a \otimes h_b)
\]

where \( g \) denotes a flattening function that converts a \( n \times n \) matrix into a \( n^2 \)-dimensional single vector.

The \textit{wide vector} \( w \) is then directly concatenated to the \textit{deep vector} \( d \). The concatenation of the \textit{wide} and \textit{deep} vectors is passed to the last fully connected layer to compute the pair score for the final output. Overall, as \((w, d)\) is the concatenation of the \textit{wide} and \textit{deep} vectors, the final output score \( Y \) for the ingredient pair \((X_a, X_b)\) is mathematically calculated as follows:

\[
Y = W_5(w, d) + b_5
\]

where \( W_5 \in \mathbb{R}^{1 \times (j^2+j)} \) and \( b_5 \in \mathbb{R}^1 \), and \( j \) is the number of hidden units in each layer.

### 4.3 Model Training Details

We train our proposed model to minimize the loss function (Mean Squared Error) which can be expressed as follows:

\[
L(\theta) = \frac{1}{N} \sum_{a,b} (y_{ab} - Y_{ab})^2
\]

where \( L \) is the computed loss function to be minimized during training, \( \theta \) are the model parameters to be trained, \( y_{ab} \) is the true score value, \( Y_{ab} \) is the predicted score value, and \( N \) is the total number of input pairs used for training. We use the Adam optimizer for our model.

### 5 Experiment

#### 5.1 Baseline Models

We first evaluated the baseline models before evaluating our proposed model. We first predicted the pairing scores by simply calculating the cosine similarity between two input ingredient vectors. We then employed the following machine learning models from the Python Scikit-learn [Pedregosa et al., 2011] package as our baseline models: Linear Support Vector Regressor, Random Forest Regressor, Extra Tree Regressor, SGD Regressor, and Gradient Boosting. Additionally, a simple version of Siamese neural networks [Koch et al., 2015] is used one of the baseline models. All these models are fitted with hyperparameters estimated by the built-in grid search.

### 5.2 Main Results

As illustrated in Table 3, the following five metrics were utilized to evaluate model performance: Root Mean Square Error (RMSE), Mean Square Error (MSE), Mean Absolute Error (MAE), Correlation (CORR), and R squared (R2). Our KitcheNette model clearly outperforms the baseline models in all metrics.

We use Normalized Discounted Cumulative Gain (NDCG@K) to evaluate the ranking performance of our model (Figure 3a) and employ ROC curve to evaluate the sensitivity of our model (Figure 3b). In terms of NDCG@K, our model outperforms all the baseline models in making accurate predictions. The ROC curve is also used to measure the classification performance of the models. As mentioned in Section 3.2, we regarded all pairings with prediction scores of 0.274 or higher as complementary pairings; pairings with lower scores were considered non-complementary. The ROC curve results demonstrate that our KitcheNette model achieves higher performance than all the other models in classifying complementary pairings.

### 5.3 Ablation Study

We performed ablation tests to evaluate each feature of KitcheNette. As illustrated in Table 4, the \textit{wide&deep} architecture and ingredient embedding of our model help improve its performance. When our model uses the cosine similarity of learned representations from the Siamese networks, it obtains the lowest performance in predicting food pairing scores. The concatenation (\textit{deep layer}) of two representations dramatically improves the performance of our model. This indicates that semantic relations need to be learned for predicting food pairing scores. Furthermore, the \textit{wide&deep} architecture that learns the relation of two ingredients further boosts our model’s performance. Also, utilizing the ingredient embedding for input vectors, instead of randomly initialized vectors, improves the model’s performance.

### 6 Qualitative Analysis

For the qualitative analysis, we considered performing experiments with actual food and obtain human feedback, but realized it would be difficult to evaluate large-scale pairing scores and is beyond the scope of our work. Instead, we performed various case studies. In addition, we provide a demo version\(^3\) of KitcheNette where anyone can retrieve the scores of ingredient pairings.

---

\(^3\)http://kitchennette.korea.ac.kr/
The prediction results of all the nine pairings were consistently high (0.33-0.45). Based on the annotated scores, we paired Case 1. We chose onion as it made the worst pairing and eight unknown pairings were also consistently high (0.29-0.42) compared to the three given known pairings. Finally, we chose onion as it made the worst known pairing with champagne and paired it with the remaining two wines. The predictions results of the two unknown pairings (sparkling wine & onion and prosecco & onion) were consistently as low as the scores of the known pairing (champagne & onion).

**Results.** In sum, our prediction results show that KitcheNette is capable of making predictions on certain pairings based on analogical reasoning which states that if A is similar to B and A forms a good pairing with C, then it is more likely that B also forms a good pairing with C. We believe that using this reasoning enhanced the performance of KitcheNette on unknown food pairings without annotated scores.

### 6.1 Finding Unknown Pairings

To demonstrate the accuracy of KitcheNette’s predictions on infrequently used food ingredient pairings, we performed a comparative analysis of the prediction results of both known and unknown ingredient pairings. As illustrated in Table 5, we chose three similar carbonated white wines (champagne, sparkling wine, and prosecco). We then calculated the score of each wine paired with a different ingredient and analyzed all the possible pairings for the three different cases below.

**Case 1.** We used the champagne & orange_twist as a given pairing for comparison since it is a well-known ingredient pairing with a high annotated score. The orange_twist & sparkling wine and orange_twist & prosecco pairings do not have annotated scores since they are uncommon pairings. Additionally, we chose two other ingredients (orange_wedge and lime_twist) that are similar to orange_twist, but are not frequently used with any of the three wines. As a result, we have one known pairing and eight unknown pairings. The prediction results of all the nine pairings were consistently high (0.33-0.45).

**Case 2.** Based on the annotated scores, we paired the wines with ingredients that are different but complement at the same time with them, to create the following three unique known pairings: champagne & elderflower liqueur, sparkling wine & cream de cassis, and prosecco & lemon sorbet. The prediction results of the remaining six unknown pairings were also consistently high (0.29-0.42) compared to the three given known pairings.

**Case 3.** Finally, we chose onion as it made the worst known pairing with champagne and paired it with the remaining two wines. The predictions results of the two unknown pairings (sparkling wine & onion and prosecco & onion) were consistently as low as the scores of the known pairing (champagne & onion).

**Results.** In sum, our prediction results show that KitcheNette is capable of making predictions on certain pairings based on analogical reasoning which states that if A is similar to B and A forms a good pairing with C, then it is more likely that B also forms a good pairing with C. We believe that using this reasoning enhanced the performance of KitcheNette on unknown food pairings without annotated scores.
recommended food ingredients that share a large number of chemical compounds with the selected ingredients, some of the recommendations did not pair well with the selected ingredients (e.g., tomato&tea, onion&cocoa, pepper&orange) for cooking and dining.

### 6.3 Discovering New Drink-Food Pairings

We also found that KitcheNette can discover new food-drink pairings, which we believe is one of the main aims of food pairing. As illustrated in Table 7, we compared our model's food&drink recommendations with those from "The Flavor Bible" [Page and Dornenburg, 2008] and "WHAT to DRINK with WHAT you EAT" [Dornenburg and Page, 2009]. We found that KitcheNette not only provides recommendations that are consistent with the recommendations of culinary experts from the books, but also recommends far more pairings than the two books.

For red wine and white wine, our model recommended a variety of meat (e.g., beef, lamb) and specific seafood ingredients (e.g., mussel, lobster, shrimp) respectively. Our model recommended authentic Japanese food ingredients to pair with sake, which shows that our data-driven learning model is also able to recommend food ingredients less common in non-Asian cuisines.

### 7 Conclusion & Future Work

In this work, we introduced KitcheNette which predicts food ingredient pairing scores and ranks them based on the predicted scores. Our model which has Siamese neural networks instead of on one-to-one pairings, which were used by our model's Siamese networks. Also, we plan to add the chemical information of food ingredients to the ingredient embeddings and use more detailed information on food ingredients from food encyclopedias. Last, we would like to use more novel and authentic recipes to help our model to recommend more versatile food ingredient pairings.

### Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the government of Korea (MSIP) (NRF-2017R1A2A1A17069645, NRF-2017M3C4A7065887)
References


