Recipes Generation Using Personal Nutrition Information

Hagit Liven Bar Yehuda
Ben Gurion University
hagitli@post.bgu.ac.il

Miri Yitshaki
Ben Gurion University
miriyi@post.bgu.ac.il

Abstract

In this work we present a novel implementation that integrates historical nutritional information into a personal recipes generation system. Our proposed architecture uses attention approach over the previous recipes that the user reacted to. It creates a unique nutrition representation for a user and a recipe, and generates custom recipes based on the collected information.

1. Introduction

Food is an important and essential part of the human life; everyone eats, cooks or uses food delivery services on daily basis. We enjoy social time with friends and family and celebrate special occasions as well as holidays, surrounding by traditional and comforting dishes, which empower the experience and provide memorable moments. Food is more than just eating or nutrition; it has social, cultural, traditional, seasonal, dietary and other valuable aspects. Food recommendation is a relatively new area of research, which has some systems deployed in real settings that focus on user preferences and are used mainly for restaurants and dishes recommendations.

When we think about making food - cooking, we usually seek for qualitative recipes that are tasty, but are also based on our preferences- such as cuisine, preparation time, ingredients and complexity.

In this evolving field of food recommendations, automatic generation of cooking recipes is an interesting and practical research problem. While traditional online cooking recipes are static and do not take into account users personal preferences, recent recipe generation systems that use advanced deep learning techniques, have the ability to personalize recipes and generate cooking instructions for a specific recipe, based on given list of user-specified ingredients.

When we cook a dish by following a recipe, we may find many constrains that need to be taken into consideration. Some of the most concerning issues in the field of nutrition today, that have crucial impact on our chosen recipe, are health and diet restrictions. According to WHO, diet and nutrition are crucial for prevention of chronic diseases [1]. The awareness of healthy food's benefits is increasing while searching for healthier substitutions becomes more relevant and essential than ever. Therefore, there is no doubt that nutrition value has to be a significant factor when considering a recipe. However, while reviewing the current approaches for recipes generation, none of the existing methods suggests fully nutrition awareness of the produced recipe. Therefore, by developing the proposed implementation, we set our goal to letting the system learn and produce recipes, based on ingredients nutrition values and actual health recommendations and standards, to allow users enjoy personalized as well as healthier recipes.

In the next sections, we comprehensively review the research that have been done in the domain of recipes recommendations in general, and in recipes generation particularly. Then we present an existing recipe generation system that was performed as a baseline to our improvement. We describe our improvement that is based on global standard nutrition score [2], discuss our evaluation concerns and challenges, and finally present a comparison of different models configurations.

2. Background

2.1. Recipe Recommendations

Content based methods are commonly used for food recommendations; Those systems use content-based attributes, where the last apply to the food item as a whole: the cuisine, the nutrition, or the spiciness of the dish. Recipes, however, are an interesting exception of content-based recommendations; They are a mix of recipe-level attributes, as well as ingredient-level attributes. That is, individual ingredients might be associated with a spiciness, nutrition, or might determine the dominant flavor or color. That is, a recipe is a vector of ingredients, each of which might have relevant weights, tags, and attributes. Recipe ratings can also be passed down the ingredients themselves.

For hybrid recipes recommendation methods, Cohen et al [3] assume that users exhibit preference for recipes, but a recipe contains explicit ingredients, each of which has a latent ingredient factor. By factorizing a user-ingredient matrix, rather than user-recipes, they produced a set of meaningful factors that corresponded to groups such as bread, beef, drinks, and barbecue. A users profile then becomes a preference across all the ingredients in all the recipes that they have rated, and a candidate recipe is a vector across the ingredient factors, which can be ranked and recommended via cosine similarity. Freyne and Berkovsky [4] built a content-based recommender by passing the ratings of recipes down to the ingredients themselves, weighted equally.

2.2. Recipe Generation

Recipe generation belongs to the field of data-to-text natural language generation [5], and the presented models use variations of RNNs architecture, which is state-of-the-art for text generation tasks and provides the ability to produce coherent and qualitative outputs. Recipe generation becomes an interesting challenge on the field of food recommendation systems as well as test-generation, and few approaches have been recently proposed in this area.

Kiddon et all [6] suggest a model that generate cooking instructions; In the proposed system, the title and ingredient list are provided as inputs, and the model produced a complete text that describes how to prepare the desired dish. Their model generates recipes as a structured collection of ingredient entities acted upon by cooking action. At a high level, the model uses a recurrent neural network (RNN) language model that encodes the goal as a bag-of-words and then generates output text token by token. It additionally stores a vector that acts as a soft checklist of what agenda items have been used so far during generation. This checklist is updated every time an agenda item reference is generated and is used to compute the available agenda items at each time step. The available items are used as an input to the language model and to constrain which agenda items can still be referenced during generation. Agenda embeddings are also used when generating item references.

Yang et al [7] use Referring expressions (REs) in order to recognize explicit ingredient references. They present reference-aware language model that generate the token at each recipe step. In their work they develop a language model that has a specific module for generating REs. A series of decisions augment a traditional recurrent neural network language model and the two components are combined as a mixture model. Selecting an entity in context is similar to familiar models of attention (Bahdanau et al.[8]), but rather than being a soft decision that reweights representations of elements in the context, it is treated as a hard decision over contextual elements which are stochastically selected and then copied or, if the task warrants it, transformed. In cases when the stochastic decision is not available in training, they treat it as a latent variable and marginalize it out.

Another noticeable work is RecipeGPT [9], a system that is based on OpenAI GPT-2 framework, and is trained on large recipes dataset and provides cooking instructions. The generation module relies on a generative pre-trained transformer GPT-2, fine-tuned on Recipe1M dataset. The authors utilize the fine-tuned model to perform two tasks: ingredient generation and instruction generation. generation model is trained and deployed to the inference engine to handle live requests. Then, the requests are encapsulated and sent to the inference engine. Lastly, MongoDB is used as a data storage to optionally save the generated outputs for future reference and inspection.

These approaches show impressive and coherent results; however, they do not address issues of personal preference, history and health considerations; Moreover, most of them cannot handle incomplete recipe details.

2.3. Recipe Health Considerations

Many people are facing the problem of making healthier food decisions to reduce the risk of chronic diseases such as obesity and diabetes, which are very relevant to what we eat. Therefore, food recommendation not only caters users food preference - but should also be able to take users health and diet into account. Correspondingly, how to build the model to balance these two components become the core problem for health and nutrient-based food recommendation. According to a latest and comprehensive survey in food recommendation systems domain[10], most methods have tried to incorporate healthiness into the recommendation process by substituting ingredients, incorporating calorie counts, generating food plans and incorporating nutritional facts. For example, Ge et al [11] simply calculated the weighted mean between the preference component and health component, where the weights are manually adjusted by the user.

A hybrid recommendation approach, which incorporates content information directly as a linear constraint to provide additional insights about the contents themselves is adopted. Ribeiro et al [12] considers more factors including nutrition, food preferences and the budget for meal recommender system for older adults.

An interesting work that proposed a personalized nutrient-based meal recommender system is Yangs Yum-Me [13], that was designed to meet individuals nutritional expectations, dietary restrictions, and fine-grained food preferences. Yum-me enables a simple and accurate food preference profiling procedure via a visual quiz-based user interface, and projects the learned profile into the domain of nutritionally appropriate food options to find ones that will appeal to the user. Yum-me operates over a given inventory of food items and suggests the items that will appeal to the users palate and meet their nutritional expectations and dietary restrictions.

2.4. Nutri-Score

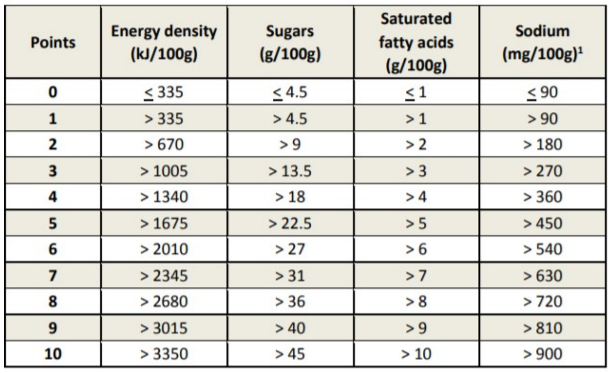
In this work, the nutrition representation is based on Nutri-Score [2] calculation.

The Nutri-Score is a nutrition label that converts the nutritional value of products into a simple code consisting of 5 letters, each with its own color. Each product is awarded a score based on a scientific algorithm. It relies on the computation of a nutrient profiling system derived from the United Kingdom Food Standards Agency nutrient profiling system (FSA score).It has also been recommended by Belgian, Spanish, German and Dutch authorities as well as the European Commission and the World Health Organization.

Figure 1. Nutri-Score label of an 'A' product

This formula takes into account the nutrients to avoid (energy value and the amount of sugars, saturated fats and salt) and the positive ones (the amount of fiber, protein, fruit, vegetables and nuts, rapeseed oil, walnut oil and olive oil). The algorithm gives points for each element in the nutrition table (per 100 g or ml)- that means bad nutrients (energy, sugars, saturated fatty acids, salt) as well as good nutrients (proteins, fiber, percentage of fruit, vegetables, nuts, rapeseed oil, walnut oil and olive oil). We then subtract the positive points from the negative ones and convert the result to the Nutri-Score table.

The number of points that is given for each nutrient value range, is presented on Figure 2.



Points	Energy density (kJ/100g)	Sugars (g/100g)	Saturated fatty acids (g/100g)	Sodium (mg/100g)[1]
0	≤ 335	≤ 4.5	≤ 1	≤ 90
1	> 335	> 4.5	> 1	> 90
2	> 670	> 9	> 2	> 180
3	> 1005	> 13.5	> 3	> 270
4	> 1340	> 18	> 4	> 360
5	> 1675	> 22.5	> 5	> 450
6	> 2010	> 27	> 6	> 540
7	> 2345	> 31	> 7	> 630
8	> 2680	> 36	> 8	> 720
9	> 3015	> 40	> 9	> 810
10	> 3350	> 45	> 10	> 900

Figure 2. Nutri-Score points per nutrients values

3. Related work

Majumder et al [14] proposed advanced and comprehensive approach that uses model attention mechanism, to provide personalize output which is based on the user preferences and history.

They were the first to consider using personalized text generation- an approach that shows successful results for user writing styles (e.g reviews)- to the problem of recipe generation, where output quality is heavily dependent on the content of the instructions such as ingredients and cooking techniques. Their main contribution is development of attention model that generates plausible and personalized recipes from incomplete input specifications, by leveraging historical user preferences.

This approach considers two different schemes to model preferences from user histories: (1) recipe interactions, and (2) previous techniques that were seen there in those recipes. During their work, the authors demonstrate better results for a model that is based on recipe interactions. The basic models input specification consists of the following elements: the recipe name as a sequence of tokens, a partial list of ingredients, and a caloric level (high, medium, low). It produces the recipe instructions as a token sequence, for the given recipe name. In the personalize approach, the model also uses historical recipe interactions of the user.

The model architecture contains two main components:

Encoder: The encoder has three embedding layers: vocabulary embedding, ingredient embedding and caloric-level embedding. First, Each token in the recipe name is embedded via vocabulary; the embedded token sequence is passed to a two layered bidirectional GRU (BiGRU), which outputs hidden states for names. Similarly, each of the input ingredients is embedded via ingredient embedding, and the embedded ingredient sequence is passed to another two-layered BiGRU to output ingredient hidden states. Next, the caloric level is embedded via calorie-level embedding and passed through a weighted projection layer with to generate calorie hidden representation.

Decoder: The decoder is a two-layer GRU with a hidden state, conditioned on previous hidden state and input token from the original recipe text. The concatenated encoder outputs are projected as the initial decoder hidden state. In order to bias generation toward user preferences, the model attend over a users previously reviewed recipes to jointly determine the final output token distribution. The model uses Bahdanau attention mechanism [8], which is applied over the encoded ingredients to use encoder outputs at each decoding time step.

As previously mentioned, two different architectures were proposed to develop user preferences model from user histories: the first is recipe interactions, and the second is techniques that appeared in the users historic recipes:

Prior Recipe Attention: The authors obtain the set of prior recipes for a user, where each recipe can be represented by an embedding from a recipe embedding layer or an average of the name tokens embedded by vocabulary. The model attends over the k-most recent prior recipes, to account for temporal drift of user preferences. These embedding representations are used in the Prior Recipe and Prior Name models, respectively.

Prior Technique Attention: The authors calculate prior technique preference by normalizing co-occurrence between users and techniques that were seen in users previous recipes space, in order to obtain a preference vector. Each technique is embedded via a technique embedding layer. Prior technique attention is calculated, and attention emphasizes the users prior technique preference.

Attention Fusion Layer: all contexts calculated are fused by concatenating them with decoder GRU output and previous token embedding. Then, the model calculates the next token probability and maximize the log-likelihood of the generated sequence, conditioned on input specifications and user preferences.

4. The Method

4.1. Motivation

Majumder original models produce good and coherent outputs; However, their work did not address the most concerning issues in nutrition today which are health and diet restrictions.

Lets take for an example two friends, John and David, who are both suffer from common health conditions: John has chronic but well-managed type 2 diabetes, and David is at risk of developing hypertension. We can assume that John would probably prefer low-sugar meals in order to control his condition, while David would try to pay attention to sodium amount in his diet.

These two friends would receive tailor-made and personalized recipes if they use the user preferences model; however, we cannot be sure the model will capture their nutrition considerations, when producing a recipe; we know that it will be aware to their previous recipes and preferable cooking techniques, but it would probably miss the crucial nutrition values.

While considering and planning the desirable improvement to the original models, we defined our goal as bringing nutrition awareness to their personalized generated recipes. The original models deal with nutrition information only by assigning one of three categories, which describe the calorie level for each recipe.

During our research, we discovered that extended nutrition information is available in the original data set. Hence, we realized that we can use this information in order to expand and thicken the nutritional representation of any recipe; instead of using simple 3 level categorical feature, we are now able to produce richer and exact details for each recipe and hopefully make choices that take into account the users previous nutrition preferences.

Implementation of the proposed architecture is available on our GitHub repository [1].

4.2. Nutri-Score calculation

In order to achieve high impact of the nutrition information for a recipe, we decided to use Nutri-Score approach as two different features. At first, we examined the effect of each feature independently, and then we used both of them for a recipe representation. Note, that due to incomplete ingredients quantities in the original recipes data set– which are needed to calculate the positive component of the original Nutri-Score–

[1] https://github.com/miriYitshaki/RecipeGenerate

we calculates only the negative part of the original Nutri-Score formula.

In order to be able to use Nutri-Score formula, we needed to collect the information from the recipes raw file. Then, we converted these values from daily values units to grams, by following the maximum recommended values for healthy people, according to the WHO [1].

4.2.1. Nutri-Score as categorical feature In this approach, we calculated the negative score by following Nutri-Score formula, for each of the bad nutrients (energy, sugars, saturated fatty acids, salt).

We summarized these values and got a number in the range of 0 to 40. Inspired by Nutri-Score scale, we set 5 categories of healthiness levels, and assigned the score to the proper category, where 0 indicates healthy recipe (low nutrients values) and 4 indicates not-healthy recipe. At last, we exchanged the original calorie level with the new calculated Nutri-Score.

Definition Let X be a nutrient, $X \in [Cals, Sugar, Fats, Sodium]$, and X_{val} be the nutrient weight in grams. Then $X_{NutriScore}$ is the Nutri-Score that match to the nutrient value, according to Figure 1. More formally:

$$X_{NutriScore} = i, X_i < X_{val} < X_{i+1} \tag{1}$$

4.2.2. Nutri-Score as OneHotVector We propose a unique representation for recipe nutrition information, which is similar to the original technique representation. We define this representation as **NutriVec**.

In this approach, we implemented an embedding method that was also used for techniques embedding. First, we collected the relevant nutritional data for each recipe from the raw data set. Then, We calculated Nutri-Score for each of the nutrients, based on formula as described on 4.2. However, instead of summarizing the calculated values to get final score as we did for 4.2.1 - we represented each bad nutrient as a sparse One Hot Vector, has a length of 11, that indicates a score of 0 to 10. We summarized these vectors to one Nutri-Vector with a length of 44, and updated the relevant source file. We also used recipe Nutri-Vectors to represent the relevant information for users; for each user, we aggregated the Nutri-Vectors recipes based on the user's interactions. At last, we implemented the Nutri-Attention relevant functions and did final adjustments on the original code, to let the model accept the new information.

4.3. Nutri-Attention

We apply attention to NutriVecs using similar attention as described on the original article, over the encoded feature. We define an attention-score function α with key K and query Q:

$$\alpha(K, Q) = \frac{\exp(\tanh(W_\alpha[K + Q] + b_\alpha))}{Z} \tag{2}$$

The Prior nutrition attention is calculated for a recipe, based on Bahdanau Attention. Than, all contexts are fused at time t, concatenating with decoder GRU output and previous token embedding. At last, token probability is calculated, and log-likelihood of the generated sequence is maximize , conditioned on input and user history, as shown on Figure 3.

For nutritional information represented as Nutri-Vec Nv in recipe $r \in R$, the Prior nutrition attention $a_t^{Nv_u}$ is calculated as:

$$a_t^{Nv_u} = \sum(\alpha(x, h_t) + p_{u,Nv}) \times Nv \tag{3}$$

Where p_u is the user's preference vector and $p_{u,Nv}$ is calculated for Nutri-Vec Nv to emphasize the attention by the users prior nutritional preference.

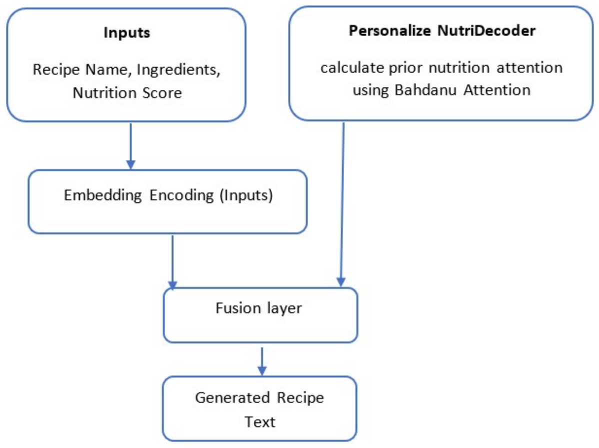

Inputs
Recipe Name, Ingredients, Nutrition Score
Personalize NutriDecoder
calculate prior nutrition attention using Bahdanu Attention
Embedding Encoding (Inputs)
Fusion layer
Generated Recipe Text

Figure 3. Sample data flow through model architecture.

5. Evaluation

To evaluate the quality and coherence of the generated recipes, we followed the quantitative metrics that were presented in the experiments and results section of the original paper. In addition to the standard metrics, the authors introduced new evaluation strategies to measure generation quality in instructional texts; they developed qualitative analysis methods, as well as novel approach to measure personalization in terms of how closely the generated text corresponds to a particular user profile.

However, during our work we noticed that required implementation instructions of the new measures were missing. The authors provide general overview of the methods however, the description was incomplete and missed the formal description of the algorithms that were used for evaluation. Moreover, no implementation of the new evaluation methods was found on the published code. Unfortunately, we were unable to recreate those custom measures, so we used the standard numeric metrics for evaluation: **ROUGE [15], BLEU [16]** and **Perplexity**. We used pyrouge [2] package to calculate ROUGE score, and NLTK [3] utilities for BLEU scores .The results are presented in detail in the results section.

5.1. Nutritional Evaluation

We tried to develop our own evaluation method, that will help us to measure and evaluate the nutrition quality of a generated recipe. Our particular goal was to examine the similarity between nutritional feature of generated recipe, to the personal nutritional profile of user, based on his historical data. We wanted to see whether Nutrition attention model produces more accurate outputs, comparing to baseline model.

5.1.1. Method The task seems to be achievable, and we defined the evaluation process as follow: First, we calculate the recipe nutritional vector of Nutri-Attention generated recipes, defined as $Nutri - Attention_{nv_r}$ vector, when r $\in$ R, and the vector is **Nutri-Vec** embedding representation of the recipe nutritional values, as described on 4.2.2. Then, we calculate the same nutritional vector of the baseline generated recipes and define it as $BaseLine_{nv_r}$ vector. Next, for each user in training set we calculate the summarized Nutri-Vec, that represents the users nutritional preferences based on his historical data and define it simply as nv_u, when u $\in$ U. At last, we planned to calculate the similarity of:

$$sim_{(nv_u, nv_r)} = ||nv_u - nv_r|| \tag{4}$$

based on the magnitude of the subtraction vector. Once we calculate the final similarity values of the pairs, we could have determined which of the methods- baseline or nutrition attention- provides the closest nutritional value for a specific user.

[2]https://pypi.org/project/pyrouge/

[3]https://www.nltk.org/_modules/nltk/translate/bleu_score.html

5.1.2. Challenges In practice, we encountered unexpected difficulties that were related to the specific experiments implementation process in the original article. We found out that the experiments mechanism the test framework- was designed for a purpose of generation recipe instructions only, with no respect to other recipes features. In the test set, the provided input is complete and given ahead: a recipe name, full list of ingredients and calorie level of the recipe. Since the nutritional value is completely depends and derived from the recipe ingredients, and the ingredients are deterministic and are provided as an input - there is no place to the model for produce its own suggestions or substitutions. In other words, this design limits the model to produce only recipes instructions.

We were surprised to discover this issue; One of the main contributions of the model that was presented in the article was complete support in partial ingredients list, when generating a recipe. We tried various approaches to manipulate the test mechanism and provide it partial ingredients list, to force it complete them in the recipe itself. For example, we recreated the initial ingredients tokens in the source files in order to hide some of the ingredients, but we were not able to significantly evaluate the impact of this change on the generated text.

In our opinion, it is possible indeed to adjust the experiments framework to support generation of other features of a recipe, rather than the instructions themselves. However, doing so in the scope of this project would required recreation of the test mechanism and would have drift us from the initial goal of the assignment - the improvement implementation.

Another concern that related to our ability to evaluate the suggested improvement, was the objective quality of the generated text. Due to limited computing resources and lack of computing power, as well as working under Googles Colab environment's restrictions and limitations, we were unable to recreate the model as it was initially presented in the article. We had to reduce the training set that were used to build the models to almost 10% of the original size. Reducing the training data has crucial impact on the training process, as well as on the generated output- while examining our experiments generated results among the different models configurations, they were significantly less coherent and understandable than the generated recipes that were presented on the article.

We believe that training the models as intended, would have bring qualitative results. The authors claim in the original article that most of the generated recipes achieve high quality results, comparing to the original recipes, based on human evaluation. Considering this assumption, we believe that the more the generated recipes are precise and coherent- the more the nutritional data will be reflected on the generated output, in the same way it has been done by the users preferences in the original implementation.

Finally, we would like to conclude by quoting the statement that claims *"evaluating the output of NLG systems is notoriously difficult"* [17]. The task of text evaluation is subjective and elusive, not deterministic and rarely can be agreed upon its quality. When evaluating systems that are design for a purpose of user interaction– either by voice or by text generation–we can consider human evaluation, which is based on well-defined surveys and questionnaires, as acceptable and reliable approach to evaluate their quality. Therefore, we believe that further and adjusted research is needed to be done, in particular in this specific field.

6. Experiments

In this work, all experiments were executed on Google Colab environment, using 12 GB RAM and cuda GPUs. We examined several values of epoch numbers and discovered that no more than 10 epochs are needed for training process, since most of the models stopped improving after 4 to 5 epochs.

Most of the settings were similar to the original paper's configurations. During our experiments we tuned the initial hyper parameters to different values. However, no improvement in model performance was indicated. We set training batch size of 32-36, hidden size of 256. Embedding dimensions for vocabulary, ingredient, recipe, techniques, and caloric level are 300, 10, 50, 50, and 5 (respectively). For attention models, we set k = 20, the 80th %-ile for the number of user interactions. We use the Adam optimizer with a learning rate of 103, annealed with a decay rate of 0.9. We also use teacher-forcing in all training epochs.

6.1. Food.com Data Set

The original Food.com data set contains 230K+ recipe texts and 1M+ user interactions (reviews) over 18 years (2000-2018) from Food.com[4]. Based on original paper, we restrict to recipes with at least 3 steps, and at least 4 and no more than 20 ingredients. We discard users with fewer than 4 reviews, giving 180K+ recipes and 700K+ reviews. in training data, the average recipe length is 117 tokens with a maximum of 256. There are 13K unique ingredients across all recipes. Due to computation limitations we needed to reduce the original training interactions set; We created a subset of 50K interactions, based on users that interact with 30 to 50 recipes. Our training-test division is 80% for training data and 20% for test.

[4]https://www.kaggle.com/shuyangli94/food-com-recipes-and-user-interactions

7. Results

We compare the results of three different configurations of the model: Baseline, user-preferences attention model and user-nutrition attention preferences model.

In baseline, no adjustments have been done on the original data; Nutritional data was not considered during training, and calorie-level was divided to original three levels. For user-preferences attention model configuration, we tried two versions of nutritional representations: with and without NutriScore. For user-nutrition attention preferences model, we also tried to see the impact of NutriVecs only, and the combination of NutriScore and Nutrivecs as described on 4.2.

The results are presented on Table 1. NR column refers to Nutritional Representation, where None indicates using original data, NV and NS are for NutriVec and Nutriscore, respectively. The model **"User Pref"** refers to original personalized model from the original paper that is based on previous recipes, and model **"Nutri Pref"** refers to the model with nutritional personal attention that uses historical nutritional information.

Model	ROUGE-L	BLEU-1	BLEU-4	PPL	NR
UserPref	**0.756**	**26.859**	12.751	12.527	None
NutriPref	0.729	26.292	12.754	12.312	NV
UserPref	0.727	25.828	12.407	12.45	NS
NutriPref	0.715	25.845	12.567	12.312	NS, NV
Baseline	0.739	26.765	**12.774**	12.278	None

Table 1. Metrics on generated recipes

8. Discussion

An initial analysis of the calculated ROUGE and BLEU scores for different model configurations, reveals that there are only small differences among the models performances; It is hard to tell which configuration is significantly better than the others. In the original paper, the authors emphasize the fact that these quantitative metrics have some limitations when evaluating generated recipes, and a careful discretion is needed when interpreting the results. In our results, we can see that the user personalized model and the baseline on the original data– with no nutritional respective– achieve the highest ROUGE.

While comparing the final results to the original paper results, we discovered that the ratio between the basic encoder-decoder model, with no penalization's perspective, remains and also repeated in our experiments. In the original paper, the baseline achieved highest ROUGE score. We believe that the quantitative metrics present better results on the baseline generated output, due to models' simplicity, comparing to complex personalized models. An explanation for this behavior was suggested on the original paper, claims that personalized models make more diverse recipes than baseline. They thus perform better in BLEU-1 with more key entities (ingredient mentions) present, but worse in BLEU-4, as these recipes are written in a personalized way and deviate from the original recipe on the phrasal level, and our results also support these findings.

As mentioned before, quantitative metrics should be taken carefully when analysing the final scores. Those results are expected; lower scores do not necessarily indicated bad quality output, but also can be caused by diverse and varied output.

Due to missing formulated instructions, we were not able to implement and recreate the qualitative methods for evaluation, as previously discussed on section 5. However, these comparisons are interesting and would have probably enlight and reveal the quality of the personalized models, as already presented by the original work. Nutritional information is implicit and latent, comparing to techniques or ingredients which are explicitly mentioned on the recipe text. Therefore, it is even more difficult to measure the effect of the collected nutritional data on the final generated output. Thus, we believe that producing proper evaluation method, or even a particular experiment that will examine the impact of the proposed nutrition representation, may also lead to interesting results- that in some cases, will elevate the generate recipe system to a practical and usable solution.

9. Conclusions

Automatic generation of personal recipes is a complicated task, but also interesting and practical research problem. It contains different aspects and concerns which can be related to personal taste, preferences, cooking skills and many more. In our research, we recognized the nutritional preferences as crucial factor that affect the generated output; While nutritional awareness and health-being interest are globally increase, a personalized and customized solutions that are able to capture those predilections are more necessary than ever before.

In this work, we implemented nutritional awareness mechanism over existing personalized recipes generation system. We adjusted an existing model to support nutritional data that was not initially considered to training process. In the proposed solution, we tried to capture and assimilate this information, using particular embedding methods, into the deep learning process in order to produce outputs that fit to users' personal nutritional history.

While testing various architectures during experiments, we faced one of the most challenging issues regard personal text generation; we realized that there is no standard for objective and well-defined evaluation method or metrics for those particular outcomes. Therefore, we defined a specific method which can be use for evaluation of nutritional value of generated recipes. However, due to design limitations of the testing framework implementation, we were not able to examine the suggested evaluation method. We believe it can be interesting direction for further research.

References

[1] WHO, "World health organization. diet, nutrition and the prevention of chronic diseases," *World Health Organization Technical Report Series*, vol. 916, pp. 1–149, 2003.

[2] J. Chantal, S. Hercberg, W. H. Organization, *et al.*, "Development of a new front-of-pack nutrition label in france: the five-colour nutri-score," *Public Health Panorama*, vol. 3, no. 04, pp. 712–725, 2017.

[3] J. Cohen, R. Sami, A. Schild, and S. Tank, "Recipe recommendation," 2014.

[4] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," pp. 111–118, 2010.

[5] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *Journal of Artificial Intelligence Research*, vol. 61, pp. 65–170, 2018.

[6] C. Kiddon, L. Zettlemoyer, and Y. Choi, "Globally coherent text generation with neural checklist models," pp. 329–339, 2016.

[7] Z. Yang, P. Blunsom, C. Dyer, and W. Ling, "Reference-aware language models," *arXiv preprint arXiv:1611.01628*, 2016.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[9] H. H. Lee, K. Shu, P. Achananuparp, P. K. Prasetyo, Y. Liu, E.-P. Lim, and L. R. Varshney, "Recipegpt: Generative pre-training based cooking recipe generation and evaluation system," pp. 181–184, 2020.

[10] W. Min, S. Jiang, and R. C. Jain, "Food recommendation: Framework, existing solutions and challenges," *IEEE Transactions on Multimedia*, 2019.

[11] M. Ge, M. Elahi, I. Fernaández-Tobías, F. Ricci, and D. Massimo, “Using tags and latent factors in a food recommender system,” pp. 105–112, 2015.

[12] D. Ribeiro, J. Ribeiro, M. J. M. Vasconcelos, E. F. Vieira, and A. C. de Barros, “Souschef: Improved meal recommender system for portuguese older adults,” pp. 107–126, 2017.

[13] L. Yang, C.-K. Hsieh, H. Yang, J. P. Pollak, N. Dell, S. Belongie, C. Cole, and D. Estrin, “Yum-me: a personalized nutrient-based meal recommender system,” *ACM Transactions on Information Systems (TOIS)*, vol. 36, no. 1, pp. 1–31, 2017.

[14] B. P. Majumder, S. Li, J. Ni, and J. McAuley, “Generating personalized recipes from historical user preferences,” *arXiv preprint arXiv:1909.00105*, 2019.

[15] C.-Y. Lin, “Rouge: A package for automatic evaluation of summaries,” pp. 74–81, 2004.

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” pp. 311–318, 2002.

[17] D. Hardcastle and D. Scott, “Can we evaluate the quality of generated text?,” 2008.