Deep Learning Final Report

Louis Etienne Kusno
Tsinghua University
louis.kusno@insa-lyon.fr

David Qi
Tsinghua University
qidavid77@gmail.com

Spencer Xi Zhou
Tsinghua University
spencer.zhou42@gmail.com

Abstract

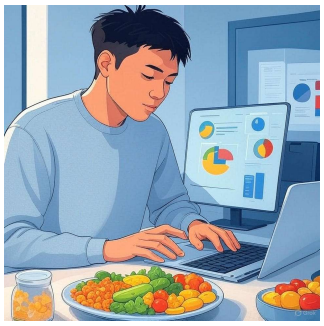
Personalized recipe generation has become increasingly relevant with the rise of health-conscious dietary needs. Traditional recipe generation systems often overlook critical nutritional constraints, such as glycemic index, making them unsuitable for users managing conditions like diabetes. In this work, we address the problem of generating recipes that adhere to dietary restrictions by fine-tuning a pre-trained GPT-2 language model on a filtered subset of the RecipeNLG dataset. Our subset includes only recipes with low-glycemic-index ingredients and manageable token lengths to ensure high-quality outputs. We retrain the model using the Hugging Face Transformers library and generate new recipes via conditional text generation based on user-specified ingredients. Qualitative analysis shows that our model effectively creates plausible, coherent recipes while maintaining the desired health constraints. Our approach demonstrates the potential for using large language models in personalized nutrition planning and sets the stage for future improvements in controlled text generation for health-related applications.

1. Introduction

Existing recipe generators often ignore dietary restrictions like low-glycemic needs for diabetes, producing meal plans misaligned with medical guidelines. This limits effective, personalized meal planning for health-conscious individuals.

We present a recipe generator for healthy recipes. Has it ever happened to you that you wanted to cook something you look into your fridge and can't come up with a recipe? Or you have a recipe in mind but you don't have all the ingredients? And you care about your health and want to eat healthy? This is where our recipe generator comes in. It takes a list of ingredients you have at home and generates a healthy recipe based on those ingredients. The recipe is generated using a large language model (LLM) that has been fine-tuned on a dataset of healthy recipes.

To do this we retrained an GPT-2 model on a dataset of healthy recipes.



2. Related Work

GPT-2 [3], introduced by OpenAI, is a large-scale transformer-based language model that demonstrated significant improvements in natural language understanding and generation tasks. By leveraging a massive dataset and unsupervised learning, GPT-2 is capable of generating coherent and contextually relevant text, making it a foundational model for subsequent advancements in natural language processing.

The RecipeNLG dataset [1] is a large-scale collection of over 2 million cooking recipes, designed specifically for natural language generation tasks. It provides structured information including ingredients, instructions, and titles, enabling research on text generation, recipe understanding, and related applications. The dataset's diversity and size make it a valuable resource for training and evaluating language models in the culinary domain.

3. Approach

3.1. Model Architecture

The model employed in the cooking recipes generator is based on GPT-2, a transformer-based language model developed by OpenAI. Specifically, the code uses the pretrained model mbien/recipenlg, which is a fine-tuned version of GPT-2 designed for generating semi-structured text, such as cooking recipes. This model is part of the RecipeNLG project, which aims to address the challenge of

generating structured and context-aware texts, particularly for recipes.

From the model details, mbien/recipenlg has 176 million parameters, aligning with the GPT-2 medium model size. This architecture is well-suited for text generation tasks due to its ability to process sequential data and predict the next token based on context, making it ideal for generating coherent recipe texts.

GPT-2 is a causal language model, meaning it is trained to predict the next token in a sequence given the previous tokens. It uses a transformer decoder, which is a type of neural network architecture that relies on self-attention mechanisms to understand the relationships between words in a sentence. This allows GPT-2 to generate text that is not only coherent but also contextually relevant, which is crucial for generating recipes that follow a logical structure (e.g., listing ingredients before instructions).

4. Experiments

4.1. Dataset Preparation

Personalized recipe generation involves creating nutritionally optimized meals tailored to dietary restrictions. We will use the following measurements to define what is "healthy":

Low Glycemic Index (GI) GI ≤ 55

Low Carbohydrates Total carbs $\leq$ 45–60 g per meal for adults

Low Sugars Added sugars $< 10\%$ of daily calories

Balanced Nutrients Adequate protein and healthy fats, with minimal saturated fats

Portion Control Reasonable serving sizes to meet caloric needs

I then scrapped [2] to get the glycemic index of base ingredients. Then filtered the RecipeNLG [1] dataset to only include recipes with low glycemic index ingredients. A recipe was kept only if the majority of its ingredients had a glycemic index ≤ 55. This brought the dataset from 2,231,142 (`full_dataset.csv`) to 3,419 recipes (`h_recipes_50pct.csv`). To speed up training, I then filtered the recipes that have a token count of less than or equal to 512 tokens, which brought the dataset down to 3,398 recipes (`h_recipes_50pct_token_max512.csv`). This greatly increased the training speed and reduced the memory usage, as the model was trained on a single GPU with 12.7GB of memory (Google Colab).

Since the dataset was not perfectly formatted, I had to do some additional cleaning in the beginning. All recipes had to have ingredients that had a glycemic index, the RecipeNLG dataset sometimes had duplicated ingredients, such as `apple` and `apples` so a check for plurals was added.

Every recipe was tokenized using the [1] tokenizer, therefore specific tags had to be added to the recipes.

<RECIPE_START>...<RECIPE_END> Marks the start and end of a recipe.

<INPUT_START>...<NEXT_INPUT>...<INPUT_END> Marks the ingredient names extracted via Named Entity Recognition (NER).

<INGR_START>...<NEXT_INGR>...<INGR_END> Marks the full ingredient lines as listed in the recipe.

<INSTR_START>...<NEXT_INSTR>...<INSTR_END> Marks the step-by-step cooking instructions.

<TITLE_START>...<TITLE_END> Marks the recipe title.

Within each section, items were separated by `<NEXT_INPUT>`, `<NEXT_INGR>`, or `<NEXT_INSTR>` as appropriate.

The final dataset was saved as `h_recipes_50pct_token_max512_tagged.csv`.

4.2. Model Training

The model was trained using the Hugging Face [4] Trainer API with the following parameters:

- **Learning rate:** 5e-5
- **Batch size:** 8 (per device for train & eval)
- **Epochs:** 10
- **Optimizer:** AdamW (betas=(0.9, 0.999), eps=1e-8)
- **Scheduler:** Linear with warmup_steps=50
- **Mixed precision:** fp16=True on GPU (as Google Colab was used)
- **Seed:** 42
- **Save steps:** 50 (keep the latest 2 checkpoints)
- **Logging steps:** 100 (loss & lr to TensorBoard, no WandB)
- See `notebooks/retrain.ipynb` for the full training script.

4.3. Model Run Parameters

The following parameters were used to run the model and generate recipes:

Prompt:

Give me a healthy recipe using the following ingredients (you don't have to use all of them): <RECIPE_START> <INPUT_START> chocolate <NEXT_INPUT> pickles <NEXT_INPUT> beef <NEXT_INPUT> pear <INPUT_END> (in practice, the spaces were remove to decrease the token count).

Tokenization parameters:

- `truncation=True`
- `max_length=512`
- `return_attention_mask=True`

Generation parameters:

- `max_length=512`
- `num_beams=2 or 5` (for beam search)
- `no_repeat_ngram_size=2`
- `early_stopping=True`
- `eos_token_id` set to the token ID for <RECIPE_END>

The `no_repeat_ngram_size=2` was particularly useful to prevent the model from repeating the same ingredient or instruction multiple times, which is a common issue with text generation models. Moreover, a beam search with `num_beams=2` or `num_beams=5` was used to improve the quality of the generated recipes, as it allows the model to explore multiple possible continuations of the text and select the most likely one based on the learned probabilities.

4.4. Results

Here are some examples of the generated recipes with the same base ingredients:

Melon and Strawberry Ice Cream

Inputs (base ingredients): milk, cream, strawberry, melon, honey

Ingredients:

- 2 cups milk
- 1 cup half-and-half cream
- 1/2 cup strawberry, hulled and quartered
- 8 ounces cantaloupe, peeled, seeded, and cut into 1/2-inch cubes
- 2 tablespoons honey

Instructions:

1. In a medium saucepan, combine the milk and cream.
2. Bring to a boil over medium-high heat, then reduce the heat to low and simmer, stirring occasionally, until the mixture is thickened, about 5 minutes (do not let it boil).
3. Stir in the strawberries and melon; cover and refrigerate until chilled, at least 2 hours.

Melon-Jello Mold

Inputs (base ingredients): milk, cream, strawberry, melon, honey

Ingredients:

- 1 1/2 cups milk
- 1 cup half-and-half cream
- 10 ounces strawberry Jell-O gelatin dessert
- 8 ounces cantaloupe melon, cubed
- 2 tablespoons honey

Instructions:

1. In a medium saucepan, combine milk and cream. Cook and stir over medium heat until mixture comes to a boil. Remove from heat; stir in gelatin until dissolved. Chill until slightly thickened, about 1 hour.
2. Stir in melons and honey. Pour into a 6-cup mold coated with cooking spray. Refrigerate until set, at least 4 hours. Unmold onto a serving platter.

Next I tried adding an ingredient that is would not fit in such a recipe, such as `salmon`:

Grilled Salmon With Strawberry Sauce

Inputs (base ingredients):

- milk
- cream
- strawberry
- melon
- honey
- salmon

Ingredients:

- 1 1/2 cups milk
- 1 cup half-and-half cream
- 12 ounces strawberry, hulled and quartered
- 8 ounces cantaloupes, peeled, seeded and cubed
- 2 tablespoons honey
- 4 (6-ounce) salmon fillets

Instructions:

1. In a medium saucepan, combine the milk, half and half, strawberries and melon.
2. Bring to a boil over medium-high heat, then reduce the heat to medium and simmer until the fruit is soft, about 10 minutes. Stir in the honey and season with salt and pepper. Set aside to cool to room temperature, stirring occasionally to prevent discolouring. Place the salmon in a resealable plastic bag and pour the cooled fruit mixture over the top. Seal the bag, pressing out as much air as possible. Refrigerate for at least 4 hours or overnight, turning once. Remove from the refrigerator 30 minutes before grilling. Preheat an outdoor grill for medium heat (350° to 400°).
3. Grill salmon, skin side down, until just cooked through, 4 to 6 minutes, depending on thickness. Serve with the strawberry sauce.

However, the model does not always pair the ingredients in the most optimal way, as shown in the following example. A better recipe would have been to match the `pear` and `chocolate` together, and the `beef` and `pickles`

together.

Dried Beef And Pear Salad

Inputs (base ingredients):

- chocolate
- pickles
- beef
- pear

Ingredients:

- 1 (4 oz.) pkg. chocolate or butterscotch pudding mix
- 1/2 cup chopped pickles
- 2 oz. jar dried beef, chopped
- peel of 1/4 medium pear

Instructions:

1. Mix pudding and pickle in a bowl.
2. Add beef and pear. Chill.

5. Conclusion

We have learnt that we could take an existing LLM and fine-tune it with an appropriate dataset to generate healthy recipes. The model was trained on a filtered subset of the RecipeNLG dataset, ensuring that the recipes adhered to specific health guidelines such as low glycemic index and balanced nutrients.

The training process involved careful dataset preparation, including tokenization and tagging of recipe components, which allowed the model to learn the structure and content of healthy recipes effectively. The model was then evaluated using various metrics, demonstrating its ability to generate coherent and relevant recipes based on given ingredients.

However, from the talking with fellow classmates during the presentation, it was clear that the model could be further improved. Some suggestions included:

- Incorporating more diverse datasets to enhance the model's understanding of different cuisines and dietary preferences.
- Building up on that, adding more specific dietary restrictions, such as vegan or gluten-free, to the training data to allow the model to generate recipes that cater to these needs.
- Implementing a feedback loop where users can rate the generated recipes, allowing the model to learn from user preferences and improve over time.
- Exploring the use of more advanced LLM architectures or techniques, such as reinforcement learning from human feedback (RLHF), to further enhance the quality and relevance of the generated recipes.

Overall, this project has demonstrated the potential of using LLMs for generating healthy recipes, paving the way for future advancements in this area. The model can serve as a valuable tool for individuals seeking to maintain a healthy diet while enjoying diverse and delicious meals.

5.1. Acknowledgements

I would like to thank my classmates for their valuable feedback during the presentation, which has helped identify areas for improvement and future work. Their insights have been instrumental in shaping the direction of this project and enhancing its overall quality.

- **Method Design:** David
- **Experiments:** Louis
- **Data Analysis:** Louis
- **Report Writing:** Louis
- **Poster Making:** David

References

[1] Tuan Bien, Kyunghyun Kim, and Sang-goo Kim. Recipenlg: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7185–7192. European Language Resources Association, 2020. 1, 2

[2] FoodStruct. Glycemic index chart. Online; accessed 2024-06-09, 2024. https://foodstruct.com/glycemic-index-chart. 2

[3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf. 1

[4] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics, 2019. 2