StepChefT5: A Multi-Step Model for Fantasy Recipe Generation

Connie Xu
Adviser: Danqi Chen

Abstract

Much prior work has been done on generating recipes from real ingredients, yet there is a growing need in the video game community to generate fantasy ingredient recipes for games. However, due to the lack of video game recipe data, it is difficult to train a fantasy recipe generation model by simply changing the data. We propose StepChefT5, a multi-step model that leverages the encoder-decoder framework of T5 and decomposes the ingredient to title and recipe task into two steps: ingredient to title, and ingredient and title to recipe. This new multi-step model shows improved learning and generation capabilities over its baseline counterpart, which generates both the title and recipe directly from the list of ingredients. A user interface for StepChefT5 can be accessed online at https://github.com/clxxu/stepcheft5.

1. Introduction

I am developing a fantasy cooking game inspired from a combination of popular cooking games like Overcooked and Papa's Games. One struggle that I had was coming up with unique, novel recipes for the game.

Figure 1 illustrates an example title and recipe[1] generated from some fantasy ingredients. The arrows between the three categories also gives hint to the types of tasks that we want to tackle: (1) ingredient to title generation, (2) title to ingredient generation, and (3) ingredient and title to recipe generation.

Automatic recipe text generation is a practical and interesting research problem that has received growing interest due to the increase in cooking recipes online. In particular, automatic recipe generation can be used to facilitate creative cooking. IBM developed a program called Chef Watson,

[1] In this paper, the word recipe will be used to refer to the set of instructions for cooking.

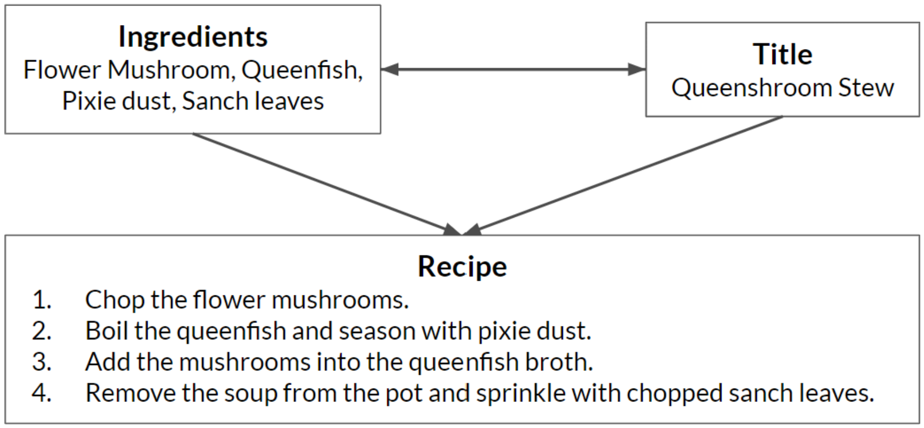

Figure 1: An example of a fantasy recipe

which generates new recipes through the analysis of ingredient pairings from different cuisine and dish types. With the help of Chef Watson, culinary experts from the Institute of Culinary Education published *Cognitive Cooking with Chef Watson*, a book which presents over 65 original recipes. The success of this book speaks to the commercial potential of the task of recipe generation.

Several approaches have been proposed and tested, like knowledge-based models [27], deep neural network models [17], and transformer-based language models [3, 18]. Similar to findings on other natural language processing (NLP) tasks, the transformer-based models were shown to be the most effective in capturing dependencies and generating fluent recipes.

Despite the growing need in the video game community to generate creative recipes for the in-game fantasy ingredients, there is still a lack of such a tool. Thus, the primary goal of this independent work is to create a tool that generates a unique title and recipe given a list of fantasy ingredients (i $\rightarrow$ tr) using a transformer-based language model. Other goals include attempting the task of generating a list of ingredients given a title, and exploring models that perform combinations of the tasks.

The amount of data that exists for fantasy recipes is extremely limited, as most video game food items have title and ingredients but no recipe instructions. Others keep their data confidential and there is no widely accessible database. Thus, rather than changing the data over which the recipe generation model is trained, we opted to change the model behind recipe generation.

We propose a multi-step recipe generation model called StepChefT5 that performs two consecutive tasks: ingredient to title generation, and title and ingredient to recipe generation. We fine-tuned seven different models using the RecipeNLG dataset on a T5-small model. Out of the seven, we selected the best title generation and the best recipe generation model for StepChefT5. We also created a baseline model which has as input a list of ingredients and outputs both a title and a recipe.

StepChefT5 performs significantly better than the baseline recipe generation model. Automatic evaluation results indicate higher BLEU scores, ROUGE scores, and METEOR scores for StepChefT5 over the baseline. Additionally, our human evaluation results conclude that StepChefT5 has better fluency, and relevance than the baseline model.

While our model shows improvement over traditional methods of recipe generation, it is still far from achieving a human level of creative generation. An analysis shows that the generated fantasy recipes often lack specificity and creativity. Much work can still be done to further this project, like augmenting the ingredient to title dataset with game recipes from existing video games, or performing further evaluation by doing named entity recognition on the generated recipes to find unwanted ingredients.

2. Related Work

In this section, we briefly explain RecipeGPT and Chef Transformer, two transformer-based language models which were trained for the task of recipe generation.

2.1. RecipeGPT

Lee et al. [18] introduced a novel web application called RecipeGPT for recipe text generation and evaluation. RecipeGPT fine-tunes a generative pre-trained language model (GPT-2) on the Recipe1M dataset [24], and provides two kinds of text generations: (1) title and ingredient to recipe generation, and (2) title and recipe to ingredient generation.

Additionally, Lee et al. developed an evaluation module to help users assess the quality of the generated texts. First, they implemented a feature which allows for highlighting of overlapped ingredients. Next, they compare the generated recipes with reference recipes by retrieving the recipe from Recipe1M with the most similar contexts. Lastly, they allow users to comment and give ratings on the generated recipes. These function as annotations which facilitate human evaluation of the generated texts.

RecipeGPT does quite well on the title and ingredient to recipe generation task as we can see in Figure 2. However, it does not provide the ability to generate a title from a list of ingredients. Users must come up with not only their own ingredients but also their own titles, which is something that we want to automate.

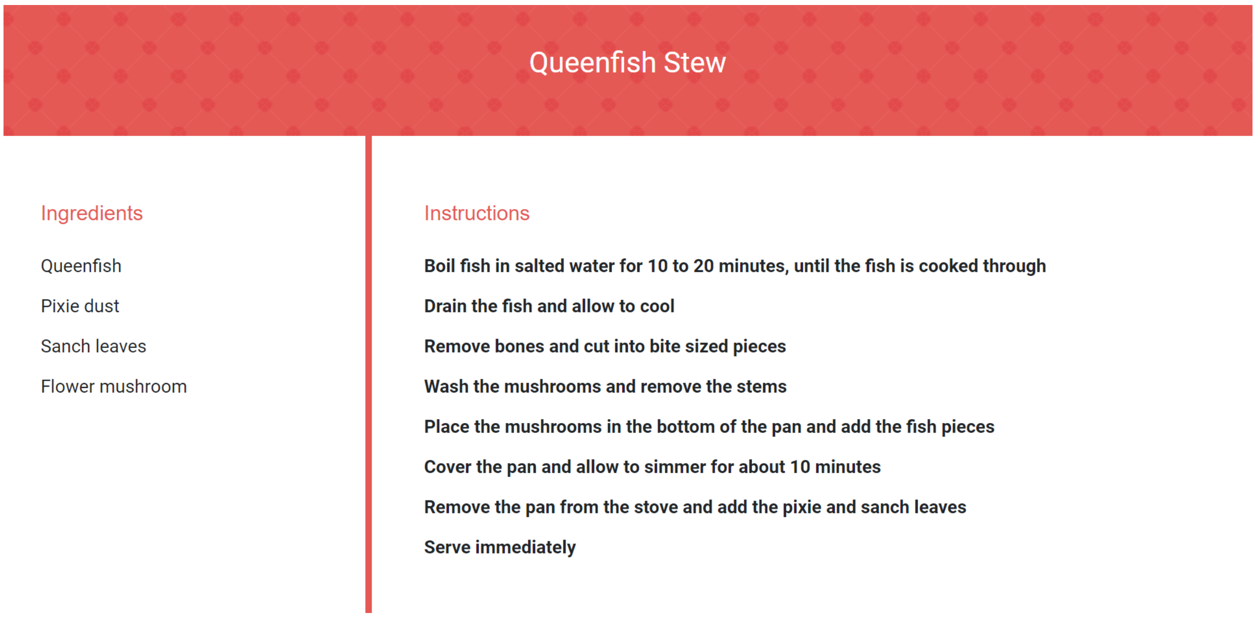

Figure 2: RecipeGPT fantasy recipe example [10]

2.2. Chef Transformer

Farahani et al. built an open-source model on HuggingFace [3] called Chef Transformer which generates recipes given a list of ingredients and a cuisine type. Chef Transformer fine-tunes a T5 model on the RecipeNLG dataset [15], which is an augmented version of the Recipe1M dataset that RecipeGPT uses. This model moves closer to what we are looking to build, since does not require title as input.

Nevertheless, it fails to generate good recipes when we input fantasy ingredients. In the recipe seen in Figure 4, the instructions involve using powdered sugar when that ingredient was not previously specified in the ingredients. Moreover, it fails to explicitly mention any of the fantasy ingredients that we specified in the ingredients list, like “zombie brains” or “dragon bran”.

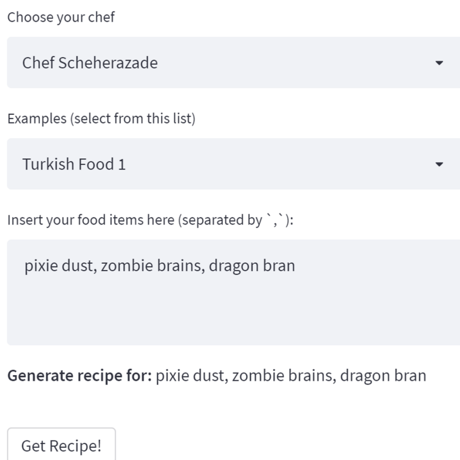

Figure 3: Chef Transformer fantasy ingredient inputs interface [2]

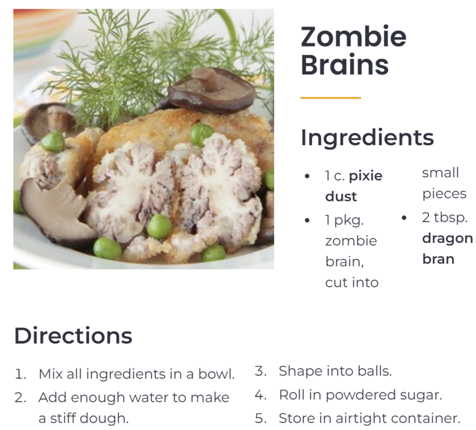

Figure 4: Chef Transformer fantasy ingredient recipe output [2]

Moreover, since the model requires generation of an image output and fantasy ingredients do not correspond to real images, the model often returns an error, thereby being of no practical use for wide-scale generation of fantasy recipes.

2.3. Problem Background Summary

RecipeGPT generates a recipe from a given a title and a list of ingredients (ti $\rightarrow$ r). The generated recipe is pretty good, but it fails to complete our desired task, which is to generate title and recipe from a list of ingredients. Chef Transformer generates both a recipe and title from a list of ingredients (i $\rightarrow$ tr). The model completes our desired task, but it fails to generate a good recipe. In this paper, we introduce StepChefT5 which completes our desired task, and generates improved recipe results.

3. Approach

To reiterate, the task is to generate a title and recipe given a list of ingredients (i $\rightarrow$ tr), and the baseline model (Figure 5) does just that.

3.1. Multi-Step Model: Title Generation and Recipe Generation

The idea behind StepChefT5 is to break this task into two steps (Figure 6). First, we generate the title from the ingredients (i $\rightarrow$ t) with a title generation model. Next, we generate the recipe from the title and ingredients (ti $\rightarrow$ r) with a recipe generation model, which we saw worked well in RecipeGPT.

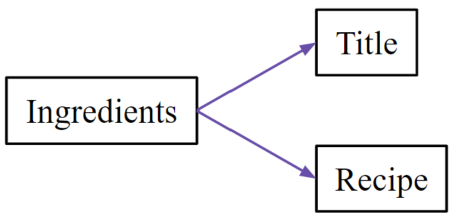

Figure 5: Baseline model
Purple lines represent the title and recipe generation model.

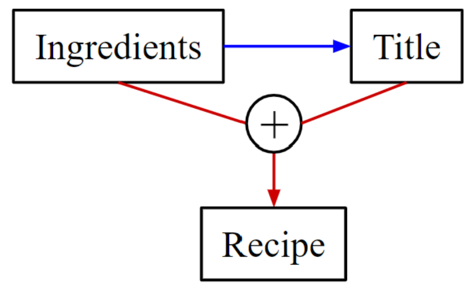

Figure 6: StepChefT5
Blue line represents the title generation model.
Red lines represent the recipe generation model.

Since StepChefT5 has a unique target output at each step of the process, it can be more focused on the task at hand. First, the model focuses on composing fantasy ingredients into a relevant fantasy title. Then, the model is able to utilize this fantasy title along with the ingredients to focus on producing a recipe. The fantasy title serves as supplementary material to the fantasy ingredients, and thereby allows for the production of a better fantasy recipe. Importantly, we prevent the model from being distracted by thinking about correct combinations of title and recipe, as that hinders the creative ability of the model.

Another benefit of this multi-step method is that we can augment the dataset with data from games.[2] Oftentimes, game developers create food items with a title and list of ingredients, but no recipe. Now, we can add title and ingredient data from games like Legend of Zelda or Genshin Impact to improve the dataset that we use to train the title generation model (i $\rightarrow$ t).

[2] See Section 7.2.

3.2. Ingredient Generation

A related task is to generate a list of ingredients from a given title. This task is intuitively more difficult than the title generation task because it needs to generate more output with less input or context. Success in this task could mean potentially creating a model which generates a list of ingredients and a recipe from just a title, which could prove useful for game developers who come up with a unique and interesting name for a food item and no ingredients.

3.3. Models

First, we created a baseline model which is trained on the task of generating a title and recipe from a list of ingredients (i $\rightarrow$ tr).

For the multi-step approach, we trained one model for each of the following three tasks:

(1) Ingredient to Title Generation (i $\rightarrow$ t)

(2) Title and Ingredient to Recipe Generation (ti $\rightarrow$ r)

(3) Title to Ingredient Generation (t $\rightarrow$ i)

We also trained models on multiple tasks in the hopes that related tasks can help the model better capture meanings and do inference. So, we had one model for each of three pairs of tasks, and finally a model trained on all three tasks.

(4) Ingredient to Title Generation, Title to Ingredient Generation (t $\rightarrow$ i, i $\rightarrow$ t)

(5) Ingredient to Title Generation, Title and Ingredient to Recipe Generation (i $\rightarrow$ t, ti $\rightarrow$ r)

(6) Title to Ingredient Generation, Title and Ingredient to Recipe Generation (t $\rightarrow$ i, ti $\rightarrow$ r)

(7) Ingredient to Title Generation, Title to Ingredient Generation, Title and Ingredient to Recipe Generation (i $\rightarrow$ t, t $\rightarrow$ i, ti $\rightarrow$ r)

A visualization of these models can be found in Figure 7.

Of the seven variations, the best title generation model and the best recipe generation model will comprise StepChefT5. We select the best ones based on their automatic evaluation metric scores.

An important thing to note is that every model that is trained on multiple tasks must use the same data over all its tasks.

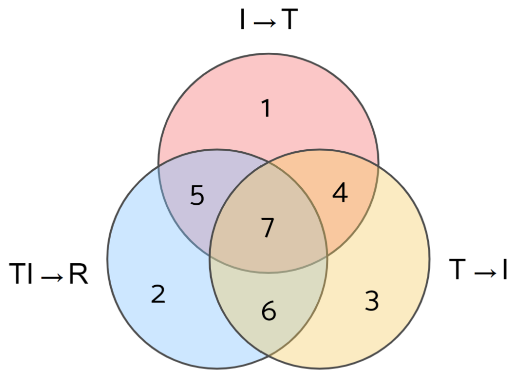

Figure 7: Visualization of the 7 models
Each color region represents a different model. For example, the orange region represents the model that was trained on both the ingredient to title and title to ingredient generation tasks.

For example, Model (6) must have complementary ingredient to title and title to ingredient data, like *recipe title generation: banana, ice cream* → *banana split* and *recipe ingredient generation: banana split* → *banana, ice cream*. If we train only on *recipe title generation: banana, ice cream* → *banana split*, and have instead *recipe ingredient generation: peanut butter* → *peanuts, sugar, salt*, the model might do surprising well when given *recipe title generation: banana, ice cream* and *recipe title generation: peanuts, sugar, salt* as test inputs. However, this will be due to the fact that the model has seen the complement in the training data, and does not imply that the model will have good performance on other actually unseen data.

4. Implementation

In this section, we describe the steps we took to build StepChefT5. First, we discuss the dataset that we used. Then, we discuss how we selected T5 as our model. Next, we go over our model training and evaluation workflow. Finally, we conclude with some notes on logging, stopping, and notation.

4.1. Dataset

For our dataset, we would ideally want to choose one with fantasy ingredients and recipes. However, due to a lack of existing data for fantasy recipes, and a lack of time to scrape video game data from websites, we opt for using a real recipe dataset.

In 2017, Salvador et al. introduced Recipe1M [24], a dataset of over 1 million cooking recipes and

800k food images. In 2019, the same group of researchers extended their image collection processes and introduced Recipe1M+ [20], a dataset of over 1 million cooking recipes and 13 million food images. Then, in 2020, a group of Polish university researchers introduced RecipeNLG, a dataset that was built on top of Recipe1M+ and enhanced with corrected and new records. RecipeNLG has a total of 2.2 million recipes.

Since our task does not require the use of images, we chose RecipeNLG, which has the most data and the most up-to-date data.

title	ingredients	directions	link	source	NER
No-Bake Nut Cookies	["1 c. firmly packed brown sugar", "1/2 c. evaporated milk", "1/2 tsp. vanilla", "1/2 c. broken nuts (pecans)", "2 Tbsp. butter or margarine", "3 1/2 c. bite size shredded rice biscuits"]	["In a heavy 2-quart saucepan, mix brown sugar, nuts, evaporated milk and butter or margarine.", "Stir over medium heat until mixture bubbles all over top.", "Boil and stir 5 minutes more. Take off heat.", "Stir in vanilla and cereal; mix well.", "Using 2 teaspoons, drop and shape into 30 clusters on wax paper.", "Let stand until firm, about 30 minutes."]	www.cookbooks.com/Recipe-Details.aspx?id=44874	Gathered	["brown sugar", "milk", "vanilla", "nuts", "butter", "bite size shredded rice biscuits"]

Table 1: RecipeNLG dataset sample entry

Out of all the columns displayed in Table 1, the ones that are relevant for our model trainings are **title**, **directions**, **NER** (the list of ingredients parsed using the named entity recognizer). We prefer **NER** over **ingredients** because it does not include the quantities. This is because we expect that users come only with ingredient ideas without predetermined quantities in mind.

To clean our data, we removed all the rows that have null values. Then, we processed the **directions** and **NER** data columns by joining all the list items into a text string which we can use as the training inputs and outputs. For example, in our model which generates a recipe given the title and ingredients (ti $\rightarrow$ r), we concatenated the title and ingredients string with a prefix and a delimiter to get the source text (Figure 8) and used the directions string for the target text (Figure 9).

4.2. Model Selection

Like the papers from Section 2, we use a transformer-based language model. There exist a wide variety of open source pre-trained transformer models, like BERT, GPT-2, and T5. The Bidirectional recipe generation: No-Bake Nut Cookies | brown sugar, milk, vanilla, nuts, butter, bite size shredded rice biscuits

Figure 8: Recipe generation task - Source text

In a heavy 2-quart saucepan, mix brown sugar, nuts, evaporated milk and butter or margarine. Stir over medium heat until mixture bubbles all over top. Boil and stir 5 minutes more. Take off heat. Stir in vanilla and cereal; mix well. Using 2 teaspoons, drop and shape into 30 clusters on wax paper. Let stand until firm, about 30 minutes.

Figure 9: Recipe generation task - Target text

Encoder Representations from Transformers (BERT [16]) model is encoder-only, meaning that it only uses the encoder part of the transformer architecture. In contrast, the Generative Pre-trained Transformer 2 (GPT-2 [22]) model is decoder-only.

Meanwhile, the Text-to-Text Transfer Transformer (T5 [23]) model uses both the encoder and decoder parts of the transformer architecture. It treats every text processing task, like translation or summarization, as a “text-to-text” task. As we can see from Figure 10, T5 takes text as input and generates text as output, which allows the model to be trained on any language task.

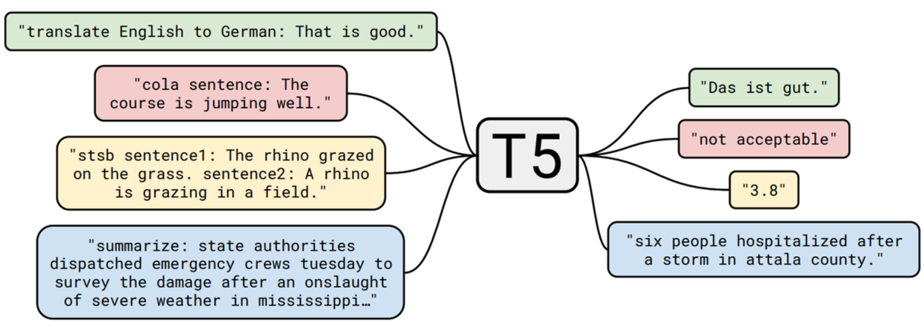

Figure 10: Diagram of T5 framework [23]
Inputs from different tasks are concatenated with prefixes to aid the model in distinguishing inputs and generating correct outputs. Adding prefixes to the input text is something that we also did for our models.

The fundamental idea behind T5 is that transfer learning and scale are useful. Training across similar language tasks can help improve performance across all tasks since there are relations between the tasks that transfer and help the model learn language better. Additionally, by training on a multitude of different tasks, the model is exposed to significantly more data, which also improves model performance. Despite the simplicity behind the idea of T5, the paper reported state-of-the-art results from using T5 when compared to other task-specific architectures.

T5's ability to involve transfer learning on a large scale fits in really well with the purposes of this project because we are trying to train models on a variety of different, related tasks. Thus, T5 is the model we selected. Specifically, we use the T5ForConditionalGeneration model because it includes a language modeling head on top of the decoder [11].

4.3. Training and Evaluation Workflow

In this section, we detail the steps in our model training and evaluation process outlined in Figure 11.

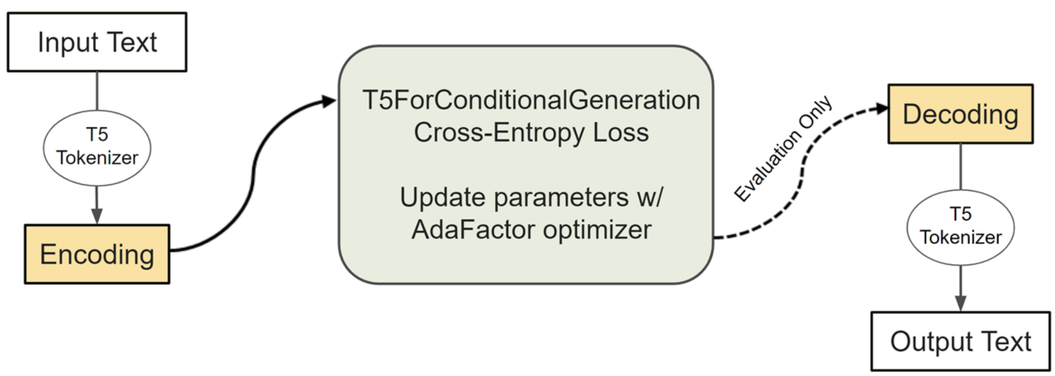

Figure 11: Model training and evaluation process

First, we feed the source and target text into a T5 tokenizer to get encodings. We encode the text in batches of 8, because that is how much our computational resources are able to handle without running out of RAM. We fix our source text encodings to be of size 256, so we truncate any inputs that are longer than 256 characters, and we pad inputs that are shorter. We chose 256 because it is the first power of two greater than the median of the source text length which is 129 for the ingredient and title to recipe generation task. Similarly, we fix our target text encodings to be of size 512. We chose 512 because it is the first power of two greater than the median of the target text length which is 351 for the ingredient and title to recipe generation task.

We proceed by feeding the encodings into the T5ForConditionalGeneration model to calculate the cross-entropy loss. We update the model parameters using an AdaFactor optimizer because that is the optimizer that T5 uses, and we use a learning rate of 3e-4 because that is recommended in the documentation [11]. This concludes the model training workflow.

To perform evaluations, we call the model to generate a decoding which we feed into a T5 tokenizer to get the output text.

All of our models are trained on only 50,000 samples per task due to resource constraints. The title and ingredient to recipe (ti $\rightarrow$ r) model training takes approximately 4 hours. For reference, RecipeNLG contains over 2 million entries, which would take 160 hours or around a week if training time scales linearly. Table 2 summarizes the relevant parameters we used in our models.

Parameter	**Value**
Batch size	8
Learning rate	3e-4
Input size	256
Output size	512
Number samples	50000

Table 2: Model Parameters

4.4. Stopping and Logging

Every training epoch, we store the training loss and calculate the validation loss. Figures 12, 13 show an example training run of the baseline ingredient to title and recipe generation (i $\rightarrow$ tr) model.

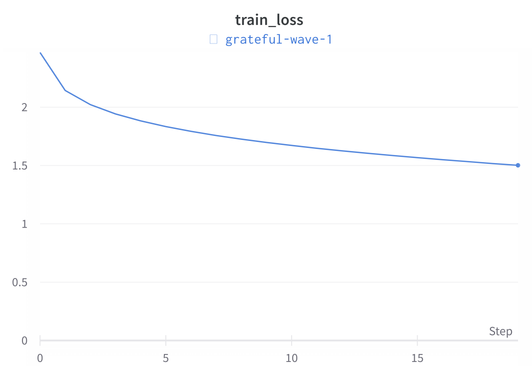

Figure 12: Ingredient to Title and Recipe: Training Loss Per Epoch

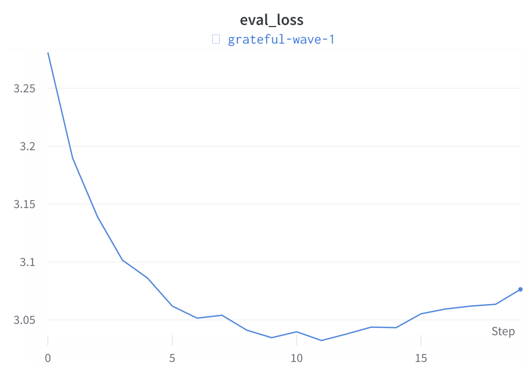

Figure 13: Ingredient to Title and Recipe: Validation Loss Per Epoch

As we can see from Figure 13, the validation loss begins to increase after around 10 epochs for the model. Thus, the final ingredient to title and recipe generation model (also known as the baseline model) which we use for comparison against other models is stopped after 10 epochs of training.

For each of the 7 model variations and the baseline model, we stop training when the validation loss increases, so the number of epochs for each model differs.

4.5. Notation

We will now introduce some notation that is used in the evaluation part of the paper. Models will be referred to by a sequence of letters and dashes and a number, like *t-i-5* or *i-tr-10*. The first group of letters refers to the model input (ingredient, title, or recipe). The second group of letters refers to the model output. The inputs and outputs tell us what task the model was trained on. For example, *t-i* refers to the title to ingredient generation task, and *i-tr* refers to the ingredient to title and recipe generation task. If a model was trained on two tasks, it will be named with something like *i-tr_t-i*. The model that was trained on all 3 tasks is named *all*. The final number refers to the number of epochs that the model was trained on.

$$\underbrace{\text{t-i}}_{\text{Task}}\_\underbrace{\text{i-tr}}_{\text{Task}}\text{-}\underbrace{\text{10}}_{\text{Epochs}}$$

Figure 14: Model variant notation

5. Results and Evaluation

We used both automatic and human metrics to evaluate the results of the title and ingredient to recipe generation task, the ingredient to title generation task, and the title to ingredient task. In this section, we explain the automatic and human metrics that we used, show the results on all three tasks, and compare them with the baseline model.

5.1. Automatic Evaluation Metrics

Our goal is to automatically evaluate for a task the quality of the generated output, and we do so with the following four metrics: BLEU, ROUGE, METEOR, and INGRE. INGRE refers to INGredient REcall, and it is a metric that we designed for this paper. For all of the metrics, scores range from 0 to 1, and a higher score signifies a higher similarity between the reference and generated output text.

5.1.1. BLEU (BiLingual Evaluation Understudy) [21] is a popular metric used for automatic evaluation of machine-translated text which computes the co-occurrence of n-grams between reference and model-generated sentences. The formula for computing n-gram precision p_n is as follows:

$$p_n = \frac{\text{number of } n\text{-grams appearing in both reference and generated text}}{\text{number of } n\text{-grams appearing in the generated text}}.$$

The overall BLEU score is a weighted geometric mean of the individual n-gram precisions, so the formula for a BLEU-n score would be as follows:

$$\text{BLEU-}n = \exp \frac{1}{n} \sum_{k=1} n \log p_n$$

For our experiments, we use a BLEU-4 score, since that is found to have the highest correlation with human judgment.

BLEU scores are typically used to evaluate translation tasks which have correct versus incorrect output answers, whereas our recipe, title, and ingredient generation tasks are more creative and open-ended. Thus, our models will have BLEU scores on the lower end of the spectrum, but this does not raise alarm, since other recipe text generation models also have low BLEU scores. For example, RecipeGPT has a test set BLEU score of 0.0858.

5.1.2. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [19] is a set of metrics used for evaluation of automatic summarization or machine translation. In this paper, we use the ROUGE$_L$ score, which measures the longest matching sequence of words using Longest Common Subsequence (LCS). Note that this differs from n-gram overlap which we see in the BLEU score because the words in the longest common subsequence do not need to be adjacent. ROUGE$_L$ is a F-measure and it is calculated using the following formulas for precision and recall:

$$P = \frac{\text{length of longest common subsequence in reference and generated text}}{\text{length of generated text}}$$

$$
\begin{aligned}
R &= \frac{\text{length of longest common subsequence in reference and generated text}}{\text{length of reference}} \\
\text{ROUGE}_L &= \frac{(1+\beta^2)RP}{R+\beta^2 P}
\end{aligned}
$$

β is a parameter usually set to be greater than 1 to favor recall.

5.1.3. METEOR (Metric for Evaluation of Translation with Explicit ORdering) [14] is another metric for evaluating machine translation. It computes the harmonic mean of unigram precision and recall on aligned unigrams. Importantly, it contains special features like synonymy matching and stemming which are not found in other metrics, and results have shown that METEOR has a higher correlation with human judgement than BLEU.

The algorithm first generates an alignment or unigram mapping between the reference and model-generated output. Define chunks as the set of adjacent unigrams in the reference and generated output. The algorithm calculates the precision, recall, harmonic mean, and penalty score as follows:

$$
\begin{aligned}
P &= \frac{\text{number of unigrams in reference and generated text}}{\text{number of unigrams in generated text}} \\
R &= \frac{\text{number of unigrams in reference and generated text}}{\text{number of unigrams in the reference}} \\
F_{mean} &= \frac{10PR}{R+9P} \\
p &= 0.5\left(\frac{\text{number of chunks}}{\text{number of mapped unigrams}}\right)^3
\end{aligned}
$$

Finally, the harmonic mean and the penalty score to are combined to get the METEOR score:

$$
\text{METEOR} = F_{mean}(1-p).
$$

5.1.4. INGRE (INGredient REcall) is a metric that we will use for the recipe and ingredient generation tasks. INGRE_r is the metric for recipe generation, and it tells us what proportion of the input ingredients are used in the output recipe. INGRE_i is the metric for ingredient generation, and it tells us what what proportion of the reference ingredients are in the generated output.

$$\text{INGRE}_r = \frac{\text{number of ingredients in input list and generated recipe}}{\text{number of ingredients in input list}}.$$

$$\text{INGRE}_i = \frac{\text{number of reference ingredients in generated output}}{\text{number of ingredients in reference list}}.$$

A low INGRE_r score would suggest that the generated recipe fails to use a significant portion of the provided ingredients. A low INGRE_i would suggest that the model fails to generate a significant portion of the reference ingredients.

5.2. Quantitative Results

In this section, we report the evaluation results on the development set, which consists of 100 samples from the RecipeNLG dataset. For each of the three tasks, we skip evaluation for the models that were not trained for the task, since those models empirically generate poor results, with incorrectly formatted output (i.e. giving a title instead of a recipe). For example, we skip the *t-i*, *i-t*, and *t-i_i-t* models for the recipe generation task.

Table 3 displays the development set scores for the recipe generation results. The *ti-r-12* model has the highest BLEU, METEOR, and INGRE scores of the four models. It has the second highest ROUGE score of 0.247, but it doesn't fall too far behind *t-i_ti-r-10* which has the highest ROUGE score of 0.251, making the difference only 0.004. Thus, it is a clear first choice for the recipe generation model in StepChefT5.

Model	BLEU-4 (%)	METEOR	ROUGE_L	INGRE_r
ti-r-12	**2.696**	**0.138**	0.247	**0.443**
t-i_ti-r-10	2.320	0.133	**0.251**	0.412
i-t_ti-r-20	1.890	0.117	0.232	0.423
all-15	0.070	0.020	0.020	0.040

Table 3: Recipe Generation Task Scores (Development Set)
The ti-r-12 model (trained only on the title and ingredient to generation task with 12 epochs) has the overall best performance on the recipe generation task. See Section 4.5 for a more detailed description of the model notation.

Table 4 displays the development set scores for the title generation results. The *i-t-5* model has the highest METEOR and ROUGE scores, and the second highest BLEU score, which is only 0.043% away from the highest BLEU score. Thus, it is safe to conclude that *i-t-5* is the best title generation model to use for StepChefT5. Interestingly, the *all-15* model outperforms the *t-i_i-t* and *i-t_ti-r* models. This is an important result because it indicates a potential synergistic effect due to the training of all three tasks which is not captured in any pairing of just two tasks.

Model	BLEU-4 (%)	METEOR	ROUGE$_L$
i-t-5	4.599	**0.184**	**0.250**
all-15	**4.642**	0.178	0.241
t-i_i-t-10	3.709	0.147	0.201
i-t_ti-r-20	2.205	0.113	0.137

Table 4: Title Generation Task Scores (Development Set)
The i-t-5 model (trained only on the ingredient to title generation task with 5 epochs) has the overall best performance on the title generation task. See Section 4.5 for a more detailed description of the model notation.

Table 5 displays the development set scores for the ingredient generation results. The *t-i-10* model has the highest BLEU, METEOR, ROUGE and INGRE scores, so it is without a doubt the best ingredient generation model out of our variations. Ultimately, none of the models that added the ingredient generation task could beat out the models trained purely on recipe generation or title generation. Thus, none of the models that are used in StepChefT5 end up using training data from the ingredient generation task, but it is nonetheless interesting to observe the results.

Model	BLEU-4 (%)	METEOR	ROUGE$_L$	INGRE$_i$
t-i-10	**3.972**	**0.197**	**0.206**	**0.246**
t-i_i-t-10	3.739	0.181	0.199	0.230
t-i_ti-r-10	3.341	0.155	0.170	0.206
all-15	0.643	0.056	0.062	0.067

Table 5: Ingredient Generation Task Scores (Development Set)
The t-i-10 model (trained only on the title to ingredient generation task with 10 epochs) has the overall best performance on the ingredient generation task. See Section 4.5 for a more detailed description of the model notation.

In summary, for StepChefT5, we use the *t-i-5* model to generate the title, and then the *ti-r-12* model to generate the recipe. It turns out that specialized training is always the best method, and anything that strays from this often leads to worse performance.

5.3. Human Evaluation Metrics

In addition to evaluating the performance of our models on real recipes, we would like to apply our models to the motivating idea of generating fantasy titles, recipes, and ingredients. Unfortunately, there is no gold standard reference for the outputs that should result from the fantasy inputs. Moreover, the tasks are meant to be creative and should have the liberty to stray from predefined outputs. Thus, we found it necessary to perform some additional human evaluation on fantasy inputs.

We created four metrics on which to evaluate the fantasy generations: fluency, relevance, format, and specificity.

5.3.1. Fluency refers to the syntactical and grammatical nature of the generated texts. Does it look like proper English? Is the grammar correct? Are the sentences complete?

5.3.2. Coherence refers to the cohesiveness and logical correctness of the generated content. Does the recipe text make use of the input ingredients in a manner that makes sense? For example, washing a pumpkin after mashing it up would not be reasonable. Does the generated title relate to the list of ingredients? For example, titling something which contains only tomatoes and lettuce as cake would not be coherent.

5.3.3. Format refers to the structure of the output. If we ask to generate a list of ingredients, does the model do so or does it generate a block of text? Is the length of the reasonable? If we asked for a title, does it produce a short phrase or a long text?

5.3.4. Specificity refers to the detail in which the output references the input. For example, a recipe that states "Combine the eggs, powdered sugar, and flour." is more specific than one which states "Combine all of the ingredients." Any recipe that explicitly contains an ingredient from the provided ingredients receives a score of 1.

5.4. Baseline Comparison

We compare the StepChefT5 model to the baseline model over the original task: ingredient to title and recipe generation. We performed automatic evaluation for the baseline and StepChefT5 model over 100 test set entries, and the results can be see in Table 6.

Model	BLEU	METEOR	ROUGE$_L$	INGRE
Baseline	1.254	0.094	0.203	**0.420**
StepChefT5	**1.357**	**0.100**	**0.222**	0.389

Table 6: Full Recipe Generation Task Automatic Metrics
StepChefT5 has higher scores in every category except INGRE.

StepChefT5 performs better than the baseline over the popular language modeling metrics like BLEU, METEOR, and ROUGE. However, it does worse than the baseline in our proposed metric INGRE. This suggests that StepChefT5 may produce recipes that are less specific and which use fewer of the provided ingredients. We confirm this finding in our human evaluation (Table 7).

Model	Fluency	Coherence	Format	Specificity
Baseline	0.893	0.810	1	**0.524**
StepChefT5	**0.976**	**0.812**	1	0.429

Table 7: Full Recipe Generation Task Human Metrics
StepChefT5 has higher or similar scores in every category except specificity.

For the fantasy recipes, we created ingredient lists with the following variety of properties to see what would happen in each case:

- long (9 ingredients) and short (1 ingredient)
- use foreign non-English ingredients (ex: chasiu)
- use made-up words (ex: pender)
- use only real words but fake ingredients (ex: flip cod)
- include incompatible foods (ex: cheesecake and shrimp)
- use non-food real item (ex: cos majors)
- use dishes rather than raw ingredients (ex: well done steak)

Overall, StepChefT5 was able to generate more unique fantasy titles than the baseline. For example, StepChefT5 produced titles like “Pimiento Pie” and “Tumbloins”. Under the same inputs, the baseline model output “Ice Cream Pie” and “Tomato-Mushroom Salad”, which are less creative.

For each case listed above, we observed the following:

- Given a long list of ingredients, both models utilize only a portion of the ingredients in the final recipe. StepChefT5 uses over half of the ingredients and the baseline using less than half. Given a short list of ingredients, StepChefT5 produces a longer and more detailed recipe output.
- When met with non-English foreign ingredients, both StepChefT5 and the baseline model resort to generating generic recipes, yet both are able to generate creative titles. In one case, the baseline model used a random unrelated ingredient not from the input out of nowhere, whereas StepChefT5 stayed more consistent by simply refraining from using the foreign vocab.
- When provided with a complete set of made-up word ingredients, StepChefT5 generated “Poppy Seed” as the title, and gave an extremely general recipe. In contrast, baseline generated “Herbed Cornbread” as the title and gave a detailed recipe which used the provided ingredients. Unfortunately, it also introduced cornbread as an ingredient when it should not have.
- When met with the task of trying to cook incompatible foods, both models similarly resort to generating generic recipes, like "Mix all."
- In the example where we use real non-food items, the baseline model incorporates the item into the recipe whereas StepChefT5 knowingly leaves it out.
- Finally, in the case where the ingredients are dishes like “well done steak”, both models create generic recipes and fail to use the ingredient in the recipe, though both incorporate “steak” in the title.

All in all, we observe that StepChefT5 is more cautious than the baseline model, though more creative with the title. It tends to use fewer of the provided ingredients, but it also does not introduce random, unprovided ingredients to the recipes.

6. Conclusion

Out of the seven model variations, the best recipe generation model is the *ti-r-12*[3] model, and the best title generation model is the *i-t-5* model. We use these two models to create StepChefT5, our multi-step recipe generation model. The best ingredient generation model is the *t-i-10* model. Unfortunately, training on the ingredient generation task did not help improve results on the other tasks, so we did not include it in StepChefT5. StepChefT5 is a promising recipe generation model, as it outperforms the baseline model under automatic as well as human evaluation metrics. It generates recipes which are relatively coherent, fluent, and correctly formatted. However, it still lacks specificity in the recipe instructions. It is more cautious than the baseline model, which has benefits for validity but loses out on some elements of detail.

7. Future Work

Much can still be done to improve StepChefT5. Here, we offer five different areas of future work.

7.1. Comparison with Related Work

We can compare StepChefT5 against RecipeGPT and Chef Transformer. We can feed the RecipeNLG test data into those models to get automatic metrics and see if StepChefT5 actually does better than those published models.

7.2. Data Augmentation

We can scrape the internet for video game recipe data and add them to our dataset. In particular, Legend of Zelda [7], Genshin Impact [6], World of Warcraft [4], Tower of Fantasy [12], Final Fantasy [9], and Pokemon [8] all have websites which can be scraped to obtain fantasy title and ingredient data. In addition, there are published cookbooks with full fantasy recipes that can be purchased as well, like *The Geeky Chef Cookbook* [5] and *Heroes' Feast: The Official D&D Cookbook* [1].

[3] Refer to Section 4.5 for a detailed description.

7.3. Further Evaluation

We can further evaluate our models using BLEURT [25], a BERT-based learned evaluation metric. It claims to yield significantly better correlation with human judgements relative to BLEU and ROUGE. We can also apply named entity recognition techniques to the generated recipe outputs to identify all the ingredient entities that were generated. This will allow us to identify the incorrectly generated ingredients so that we can calculate an ingredient precision score for the recipe generations.

7.4. Change Optimizer

One idea is to increase the learning rate of the optimizer to 1e-3, which is what the original authors of the T5 paper used as their learning rate for fine-tuning [23]. Another idea is to use Adafactor without setting a fixed learning rate, since Adafactor is a memory-efficient optimizer that is meant to adapt to the specified scale parameter and relative step value. [26] A third idea is to use Adam if one has enough computational power. Anil et al. [13] demonstrated that Adam produces higher test accuracy than Adafactor on the masked token prediction task (predicting masked tokens based on surrounding context). The downside is that it takes more memory.

7.5. Increase Computational Power

Access to better machines or more efficient computation methods can help us train on more samples. Training on the full two million samples would surely improve model performance. Similarly, training on a T5-large model rather than the T5-small model would also provide benefits.

8. Acknowledgements

First, I would like to thank my advisor, Professor Danqi Chen for her invaluable guidance and support for my project. I would also like to thank the IW seminar TAs Jacqueline and Jenny for helping me with the toughest bugs and for giving me advice on writing my paper. I am also grateful to my fellow seminar classmates from *COS IW04: Hands-on Deep Learning for Language Understanding* for providing weekly feedback. Next, I would like to give special mention to my cooking game buddies Jayson, Cataleya, and Sophie for inspiring me to work on this project. Finally, I would like to acknowledge SEAS Funding for providing me with financial support and access to better computational resources.

References

[1] "Brunsiwck stewed wereboar - dnd inspired cooking." [Online]. Available: https://www.quartoknows.com/books/9781631067105/The-Geeky-Chef-Cookbook.html

[2] "Chef Transformer Streamlit." [Online]. Available: https://share.streamlit.io/chef-transformer/chef-transformer/main/app.py

[3] "Chef Transformer (T5)." [Online]. Available: https://github.com/chef-transformer/chef-transformer

[4] "Cooking recipes - wowpedia." [Online]. Available: https://wowpedia.fandom.com/wiki/Cooking_recipes

[5] "The geeky chef cookbook." [Online]. Available: https://www.quartoknows.com/books/9781631067105/The-Geeky-Chef-Cookbook.html

[6] "Genshin impact cooking recipes list." [Online]. Available: https://genshin.gg/cooking

[7] "Legend of zelda: Breath of the wild cookbook (backend)." [Online]. Available: https://github.com/lexbonder/zelda-cookbook-backend

[8] "Pokemon quest mankey." [Online]. Available: https://rankedboost.com/pokemon-quest/mankey/

[9] "Recipe (final fantasy xv)." [Online]. Available: https://finalfantasy.fandom.com/wiki/Recipe_(Final_Fantasy_XV)

[10] "RecipeGPT: Generative Pre-training Based Cooking Recipe Generation and Evaluation System." [Online]. Available: https://recipegpt.org/

[11] "T5." [Online]. Available: https://huggingface.co/docs/transformers/model_doc/t5

[12] "Tower of fantasy cooking recipe." [Online]. Available: https://findurthing.com/tower-of-fantasy-cooking-recipe/

[13] R. Anil *et al.*, "Memory-efficient adaptive optimization for large-scale learning," *CoRR*, vol. abs/1901.11150, 2019. Available: http://arxiv.org/abs/1901.11150

[14] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–72. Available: https://aclanthology.org/W05-0909

[15] M. Bień *et al.*, "RecipeNLG: A cooking recipes dataset for semi-structured text generation," in *Proceedings of the 13th International Conference on Natural Language Generation*. Dublin, Ireland: Association for Computational Linguistics, Dec. 2020, pp. 22–28. Available: https://www.aclweb.org/anthology/2020.inlg-1.4

[16] J. Devlin *et al.*, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. Available: http://arxiv.org/abs/1810.04805

[17] C. Kiddon, L. Zettlemoyer, and Y. Choi, "Globally coherent text generation with neural checklist models," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 329–339. Available: https://aclanthology.org/D16-1032

[18] H. H. Lee *et al.*, "Recipegpt: Generative pre-training based cooking recipe generation and evaluation system," *CoRR*, vol. abs/2003.02498, 2020. Available: https://arxiv.org/abs/2003.02498

[19] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. Available: https://aclanthology.org/W04-1013

[20] J. Marín *et al.*, "Recipe1m: A dataset for learning cross-modal embeddings for cooking recipes and food images," *CoRR*, vol. abs/1810.06553, 2018. Available: http://arxiv.org/abs/1810.06553

[21] K. Papineni *et al.*, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318. Available: https://aclanthology.org/P02-1040

[22] A. Radford *et al.*, "Language models are unsupervised multitask learners," 2018. Available: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf

[23] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *CoRR*, vol. abs/1910.10683, 2019. Available: http://arxiv.org/abs/1910.10683

[24] A. Salvador *et al.*, "Learning cross-modal embeddings for cooking recipes and food images," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3068–3076.

[25] T. Sellam, D. Das, and A. P. Parikh, "BLEURT: learning robust metrics for text generation," *CoRR*, vol. abs/2004.04696, 2020. Available: https://arxiv.org/abs/2004.04696

[26] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," *CoRR*, vol. abs/1804.04235, 2018. Available: http://arxiv.org/abs/1804.04235

[27] L. R. Varshney *et al.*, "A big data approach to computational creativity: The curious case of chef watson," *IBM Journal of Research and Development*, vol. 63, no. 1, pp. 7:1–7:18, Jan. 2019. Available: https://doi.org/10.1147/jrd.2019.2893905

Appendix

Ingredients:
flower mushroom, queenfish, pixie dust, sanch leaves

Model: Baseline
Title: Queenfish Stuffing
Recipe:
Cook mushroom in boiling water until tender. Drain and remove skin. Place fish in a large bowl. Add pixie dust and sanch leaves. Toss to coat. Add mushroom mixture to fish and toss to coat. Bake at 350° for 30 minutes.

Model: StepChefT5
Title: Stollen
Recipe:
Mix all ingredients together and bake at 350° for 30 minutes.

Figure 15: Baseline vs. StepChefT5 Example 2
In this example, StepChefT5's recipe instructions are much less detailed than the baseline. However, one interesting phenomenon is that it generates a non-English word for the title. This is an exciting result, which probably also contributed to the subpar recipe text generation.

Ingredients	Baseline	StepChefT5
fungi, seaweed, salt, ginger	Wild Wild Wilderness Mix all ingredients together. Store in covered container.	Pesto Sauce Mix all ingredients together and cook on low heat until thick.
crustacean, eggs, sugar, milk	Egg Custard Mix all ingredients together. Pour into a 9 x 13-inch pan. Bake at 350u00b0 for 30 minutes.	Chess Pie Mix all ingredients together. Pour into unbaked pie shell. Bake at 350u00b0 for 45 minutes.
pumpkin	Pumpkin Pie Mix all ingredients together and pour into a baked pie shell.	Pumpkin Pie Mix all ingredients together and pour into unbaked pie shell. Bake at 350u00b0 for 45 minutes.
soy sauce, salt, vinegar, marsupial	Marinated Marinated Marinated Marinate for 2 hours.	Marinated Vegetables Mix all ingredients together and marinate for at least 2 hours.
sugar, peppercorn, vingear, pears	Pears With Peppercorns Cut vingear into thin strips. Place in a large pot. Add sugar and peppercorn. Cook until pears are tender. Add veear and cook until tender. Add pears and cook until tender.	Pears With Peppercorn Cut pears into quarters. Place in a large saucepan with the scalded peppercorn. Cook over medium heat until pears are tender, about 30 minutes. Drain.
cos majors, pepper, paprika, soy sauce, rice, ginger, broccoli, egg, bse student tears	Broccoli-Cheese-Cheese Casserole Combine all ingredients except egg and bse student tears. Pour into a greased casserole. Bake at 350u00b0 for 30 minutes. Remove from oven and sprinkle with bse Student tears.	Broccoli-Cosma Rice Combine broccoli, ginger, pepper, egg and soy sauce. Mix well. Add rice and stir. Cover and chill.
flour, ice cream, pimpernickel, peppers	Ice Cream Pie Mix flour, pimpernickel and peppers. Add ice cream. Mix well. Pour into baked pie shell. Top with meringue. Bake at 350u00b0 for 30 minutes.	Pimiento Pie Mix all ingredients together and pour into unbaked pie shell. Bake at 350u00b0 for 30 minutes.
ketchup, high protein yogurt, well done steak	Barbecue Steak Mix all ingredients together. Place in a shallow baking dish. Bake at 350u00b0 for 1 hour.	Quick And Easy Steak Dip Mix all ingredients together and serve with crackers.
snapple, green bananas, mofongo	Banana Split Salad Cut up the snapple and bananas. Add mofongo. Mix well. Chill.	Banana Snapple Slice bananas and place in a bowl. Add a little water. Mix well. Add a little water. Cook until tender.
tajin, jajiang mian, seafood boil, carrots	Tajin Jajin Mix all ingredients together. Chill.	Tajiang Mian Salad Mix all ingredients together and chill.
grapefruit, late meal cookies, finger limes, king crab	Grapefruit Salad Mix grapefruit, cookies and limes together. Add crab and chill.	Crab Cake Mix all ingredients together. Bake at 350u00b0 for 30 minutes.
cheesecake, shrimp, potatoes, ice cream	Shrimp And Potato Casserole Mix all ingredients together. Bake at 350u00b0 for 30 minutes.	Shrimp Dip Mix all ingredients together and serve with crackers.
cupcake, chasiu, beef tartare, sea urchin	Chutney-Chicken-Chicken Dessert Prepare cupcake mix according to package directions. Pour into a 9 x 13-inch pan. Sprinkle with chasiu and beef tartare. Cover with foil. Bake at 350u00b0 for 30 minutes. Remove foil and cool. Top with urchin. Serve with ice cream.	Chasiu Cupcake Prepare cupcakes as directed on package. Place in cupcake pan. Sprinkle with chasiu and cover with a layer of tartare. Sprinkle with urchin. Bake at 350u00b0 for 45 minutes.
huevos rancheros, ceviche, pancake	Colored Ceviche(Serves 8) Wash and dry the grapes. Cut the grapes into small pieces. Add the pancake. Cook until thick. Serve with a salad.	Rancheros And Rancheros Cut Rancheros in half lengthwise. Cut each rancheros into quarters. Place in a large skillet. Add a little water. Cook until tender. Drain and add a little water. Add a little water if needed. Serve with a pancake.
crepe, syrup, salt, sugar	Caramel Icing Mix all ingredients together and cook until it forms a soft ball in cold water. Remove from heat and add 1 teaspoon vanilla.	Caramel Corn Mix all ingredients together and cook until it forms a hard ball in cold water. Pour into a large roaster and let it cool.
okonomiyaki, bittermelon, garlic powder, eggs	Oven-Fried Eggs In a large bowl, combine okonomiyaki, bittermelon and garlic powder. Add eggs and mix well. Pour into a greased baking dish. Bake at 350u00b0 for 30 minutes.	Okra Mix all ingredients together and cook until thick.
flower mushroom, queenfish, pixie dust, sanch leaves	Queenfish Casserole Mix all ingredients together. Bake at 350u00b0 for 30 minutes.	Queenfish Mushrooms Wash and drain mushroom stems. Place in a large bowl. Add 1 tablespoon of pixie dust and a few drops of water. Cover and refrigerate for at least 2 hours.
trank, treen horn, flip cod, oil	Hot Dogs In a large skillet, brown trank and horn in oil. Add trank and horn. Cook until trank is done. Serve with a sour cream.	Hawaiian Fish Cut fish into bite size pieces. Dip in oil and then in a mixture of the treen horn and flip.
eggplant, dragon tears, alfalfa, sorghum grass	Eggplant Salad Peel eggplant and cut into small pieces. Cook in boiling salted water until tender. Drain and rinse in cold water. Add alfalfa and sorghum. Add to eggplant and stir.	Eggplant And Rice Boil eggs and mash. Add alfalfa and sorghum grass. Cook until done.
basil leaves, zombie brain, troll thimblets, maki, rum	Tomato-Mushroom Salad Mix all ingredients together.	Tumbloins Cut each piece of lettuce into 4 pieces. Place in a large bowl. Cover with thimblets and maki. Cover with rum. Let stand for 1 hour. Drain.
pender, stamble, nork, fanty	Herbed Cornbread Cut cornbread into small pieces. Add stamble and nork. Add fanty. Cook until done.	Poppy Seed Mix all ingredients together. Add enough water to make a thick paste.

Table 8: Example Fantasy Ingredient Inputs and Baseline and ChefStepT5 Outputs. This is the data on which we performed human evaluation.

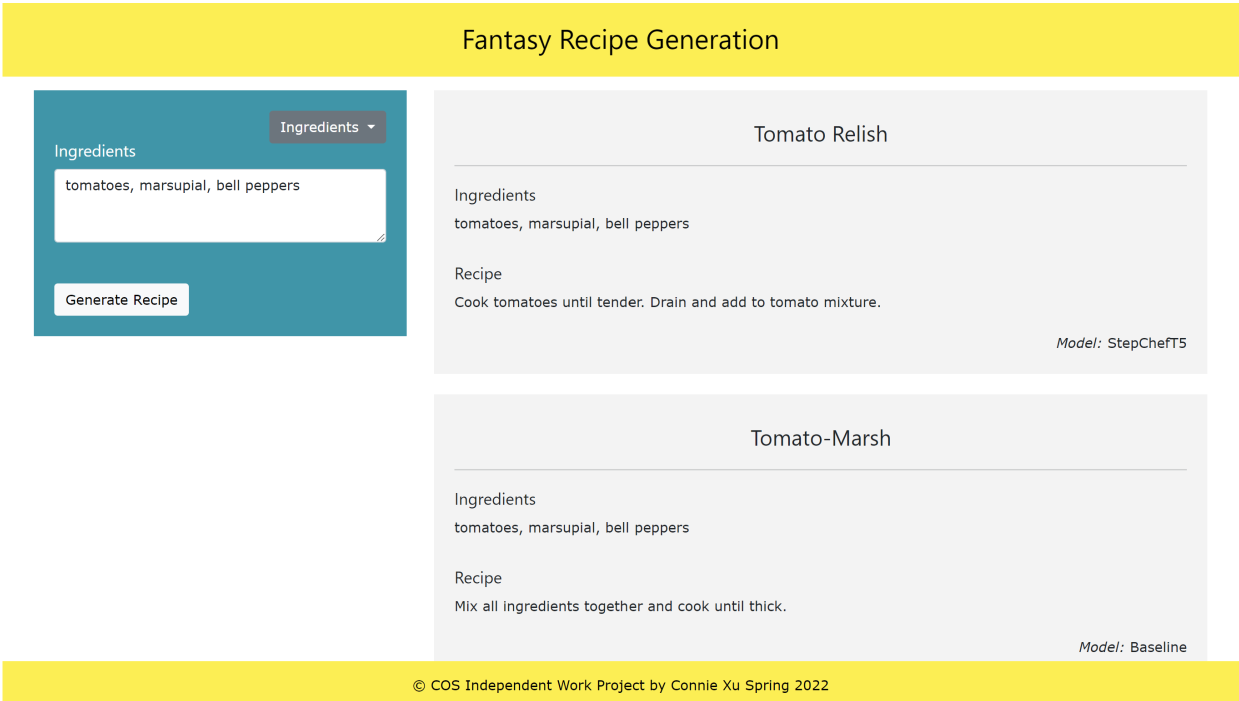

Figure 16: UI Demo 1

Generate a title and recipe given ingredients using both StepChefT5 and the baseline.

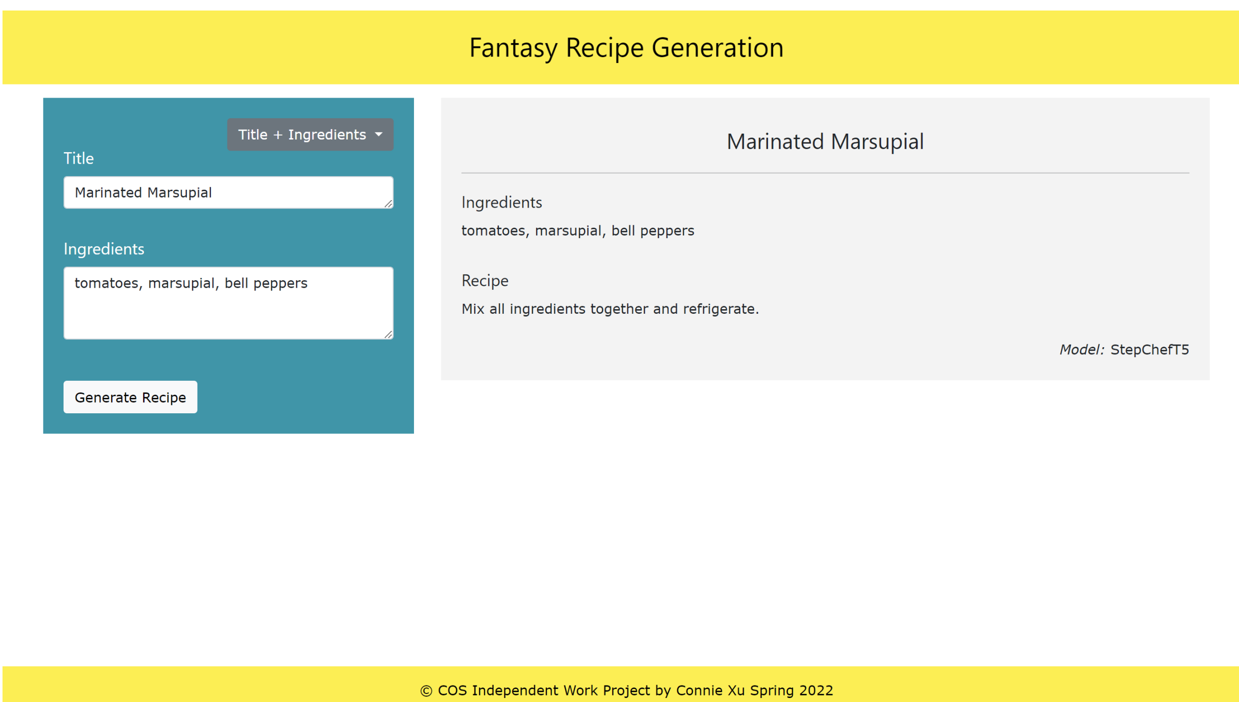

Figure 17: UI Demo 2

Generate a recipe given a title and ingredients using StepChefT5.