Transforming Cooking

William Braga, Alexander Schott

Abstract

As food is one of the most significant human cultural artifacts, the study of food as text, in the form of recipes, is rich for language model development and study. Many models have been developed to generate recipes, and they have always had at least some problems with producing recipes that are useful to humans and not simply approximations of the way we phrase cooking instructions. Oftentimes, computer generated recipes contain instructions that are reasonable sentences in terms of syntax, but impossible instructions for a human to execute. In this project three different language models were developed to generate food recipes. Text generated by these models was annotated to determine the plausibility of the instructions generated by the language model, as a means of studying the strengths and weaknesses of the various language models in the study. The study found that the char level recurrent neural network had problems with repeating itself and replicating the form of ingredient lists. The word level recurrent neural network was particularly prone to generating impossible instructions. Finally while the base GPT-2 model was found to be excellently equipped to predicting the next word in a food recipe, fine-tuning the GPT-2 model allows it to produce recipes that are near perfect in their ability to produce instructions that are not self-contradictory. The source code is at https://github.com/alexander-schott/transforming-cooking.

1 Introduction

Food is an extremely important human cultural artifact. Furthermore, the tradition of using recipes to share these cultural artefacts is a written tradition with unique syntax and word choices designed to convey information in a manner that is immediate and unmistakable to chefs. Use of language processing techniques for better understanding the processes of making food requires language models equipped for the unique challenges of food. With this project we set out to achieve two tasks in pursuit of building stronger language models for cooking instruction.

The first goal is to utilize the strengths of transfer learning to build more effective language models. Transfer models like GPT-2 have the advantage of condensing language data very compactly into memory. This property allows GPT-2 to fit much more of the text into its context window. In a recipe all previous steps inform the next step, and all information is expected to relate back to the list of ingredients. It was the expectation of the researchers that for this reason, along with other inherent strengths, transfer models would produce text that is significantly more coherent that older approaches to modeling recipes like word embeddings and long short-term memory models.

A second goal of the project was to determine whether a form of extrinsic evaluation of recipe language models could be developed to provide insight into the strengths of the model not well represented by intrinsic evaluations. While many language models have been developed to create food recipes, they all tend to produce recipes that have serious errors in them. It is common to find in computer generated recipes steps that are not possible for human beings to recreate in a kitchen. These errors are easy for human beings to notice but are difficult to measure using intrinsic measures like accuracy and perplexity. We set out

to find a measurement that can capture these issues by designing a method for annotating recipes generated by the language models for instructions that are impossible, or otherwise flawed.

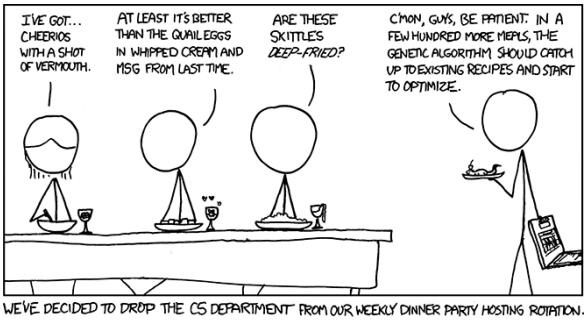

Figure 1: https://xkcd.com/720/

2 Related Work

In 2016 IBM released a repurposed Watson designed to assist home chefs in the kitchen [1]. The model used Watson's cutting-edge document searching power to recommend to chefs combinations of four ingredients that the model is confident will work together. While the model was not able to create full recipes, its unique premise was able to inspire some professional chefs to release a cook book co-authored with Chef Watson.

In 2017 Tome Brewe proposed the question on github "Do androids dream of Cooking?" [2]. His character level-rnn rose to semi-viral fame by generating recipes that captured all the formatting characteristics of food recipes while generating hilariously impossible recipes. The popularity of the generated recipes outside of the language technology sphere indicates that computing use in the kitchen is an exciting prospect for many chefs [3].

Natural language processing of food extends beyond generating novel recipes. Another innovation has been to use vector semantics and machine learning to find hidden patterns in recipes related to unique human cultures [4]. Svistova notes the popularity of recipe analysis for ingredient recommendation, recipe search, and recipe generation, but Svistovo continues to propose the use of natural language processing for relating the ingredients of recipes to the culture from which they come. The project successfully used neural networks and a SimRank algorithm to predict the cultures from which recipes came and to identify similarities in food cultures between many different world cultures.

Another trend in the foodie sphere of natural language processing has been the inclusion of image processing techniques. A team from Universitat Politècnica de Catalunya, with Facebook AI research, created a model that generates from an image of food a recipe to recreate the dish in the image [5]. This team leveraged transformers in their language model to great effect.

3 Data

The dataset used for this project takes the form of over 70,000 recipes gathered from the Meal-Master website. Meal-Master is a tool designed to help chef's find new recipes, and the dataset has been compiled and formatted by the Meal-Master community to be used by the tool. The recipes come from a variety of sources, but they come primarily from online food blogs like Bon Appetit. Recipes typically include a title, a food categorization, information about the source, and of course ingredients and instructions. Sometimes recipes also include information about the cultural background of the dish.

The recipes are stored in thirty-three one megabyte files. Storing the recipes in multiple files makes it easy to avoid loading all of the recipes into memory at once, which was important as moving the recipes into the form factor to fit the models proved to require immense amounts of data. To clean the data some information specific to the Meal-Master system was edited out of the dataset. In this process we replaced some long dividing strings with shorter end characters to separate the recipes.

Type	Chars	Words	Lines	Recipes
Total	39485884	7909456	1053963	72577
Train	22744193	4567616	607559	41818
Dev	8410209	1655639	220864	15489
Test	8331482	1686201	225540	15270

Figure 2: Dataset Splits by Type

Statistic	Value
Number of Documents	33
Average Document Size	1 MB
Number of Unique Characters	125
Number of Unique Words	70199
Number of Unique Common Words (Frequency > 4)	16198

Figure 3: Dataset Statistics

4 Models

The new model built for this project was the GPT-2 124MB model fine-tuned on our collection of recipes. With the beauty of pre-trained transformer learning, very little was required in terms of choosing parameters and designing the model architecture. The model was put through a training regime where it would spend five hundred epochs learning each of the nineteen recipe bearing files. The model would repeat this process four times so that it would ultimately spend two thousand epochs learning the ~60,000 training recipes in the training files. For text generation purposes the temperature setting of 0.7 was chosen as it has been reported to produce sound results for the GPT-2 model.

Two more models were built for this project to compare with the GPT-2 model. A recurrent neural network trained at the word level, and another trained at the character level.

The main intrinsic evaluation we decided to measure for this project is perplexity. As accuracy will always be low for predicting text when there are so many thousands of choices, perplexity is a robust way of determining how well the language model is representing the chosen text. The length of sequence that the char-rnn and word-rnn could evaluate is limited to their sequence length. In order to make a fair comparison with GPT-2, the length of sequences that our transformer model would evaluate for perplexity would be a text of the same number of words that the word-rnn is limited to. At the end perplexity was normalized to sentence length, and an average perplexity for sequences selected from our training partition was calculated.

In addition to intrinsic evaluations, annotations are an important aspect of this project. Annotations were performed by the researchers on the recipes generated by the models. The goal of the annotations is to determine how well the models are able to capture the requirement of food recipes where the instructions must be possible to be executed by a human. A more in-depth explanation of our annotation methodology is offered in the results section.

5 Baseline

The three models we built for this project were compared to each other. In addition, the pre-trained GPT-2 model was evaluated on our evaluation split of recipes for perplexity. The pre-trained model was not extrinsically compared to the new models as it does not produce recipes.

6 Experimental Setup

Sixty percent of the data was used to form the training partition, another twenty percent was set aside for the development partition used to choose hyperparameters, and the final twenty percent was used as the evaluation dataset.

7 Recipe Generation

Here are results from the char-rnn:

“1 egg
1 tb sugar
1 ts vanilla
2 ts baking powder
1/2 ts ground coriander
1 ts ground coriander
1 ts thyme
1 ts salt
1/2 ts cornstarch
1 ts salt
1 ts vanilla

mix the salt and salt in a large bowl. set aside. in a large skillet over medium heat until the chicken is tender. add vanilla. pour into a bowl. cover and let rise in a warm place until the squash is tender, then then beat the corn flavor in the center of the bottom of the remaining sauce, then add the cheese and toss to coat. serve with a spice but not

completery before serving, cover and simmer 1 minute or until the skins of the pan and place on a heated cookie sheet. bake at 375 degrees f. for 20 minutes, or until the chicken is cooked, the sauce thickens. place on a serving dish. serve hot or cold."

Here is a result from the word-rnn:

"1/4 c diced nuts
1/3 c salt
1 c water
1/4 c flour
4 oz ground red sauce
1 ts salt
1/2 ts paprika
1/2 ts dried thyme -- minced
1/4 ts dried thyme -- well

preheat oven to 350 degrees f. over chicken . reduce pan and toss (
sprinkle with
salt and pepper . peel meat with rice , in small skillet . stir turn
in ' . cook slightly about 10 minutes or until browned but you hollow
up to stand to 4 minutes . add eggs and baking processor . cut batter in
baking sheet . fill 1/2 egg (
directions ."

And here is a result from the GPT-2 model:

"Yield: 4 Servings

1 lb Ground turkey; cooked
1/2 c Ketchup
1/2 c Celery; chopped fine
2 Onions; chopped fine
2 Carrots; chopped fine
2 Stalks of celery; chopped fine
2 tb Tomato paste
1 tb Worcestershire sauce
Salt & pepper
3/4 c Rice; cooked
1/2 c Ketchup

Recipe by: Cooking without Ketchup Cook turkey in a heavy dutch
oven. Add onion, celery, carrot, and Worcestershire sauce. Place in
oven and cook, covered, 30 minutes. Add rice. Bring ketchup and serve
with salt and pepper."

And for comparison, here is text generated by GPT-2 that has not been fine-tuned to a specific task.

"I've seen the dogs.

I've seen the dogs.

I've seen the dogs.

I've seen the dogs.<|endoftext|>The O'Keefe-era civil rights group has filed a $1.5 million lawsuit against the State Department alleging that the department's use of "third-party consent" in the handling of the Benghazi consulate's emails was unlawful."

While the text generated by the models is immediately recognizable as food recipes, the GPT-2 model appeared to be the most consistent in providing text that was intelligible and easy to parse. This easiness of parsing is reflected in the good scoring of the GPT-2 model in our extrinsic evaluations.

8 Annotations

After the models were tuned, human annotations were used to evaluate the models. The annotations were performed by the two researchers. As the goal of the annotations was to determine the rate at which the models erred by producing instructions that were impossible, the researchers determined three categories of error that they would annotate. The first is redundant ingredients. The researchers determined that a recipe repeating the same ingredient more than once in its list of ingredients is a breach of Grice's speech maxim of brevity, and is a mistake not befitting a recipe produced by a human. The second error that would be annotated was described as an impossible sequence. An impossible sequence is an instance where a step in the instructions is impossible because it can not be performed in the context of the previous instruction. As recipes are meant to be a series of easily followable instructions, if the order of instructions prevents the recipe from being followed this is an inexcusable error. The final type of error that would be annotated is impossible instructions. An impossible

instruction is the gravest error, we determined, the language models could make. An impossible instruction is an instruction that is on its own nonsensical and impossible for a human to reproduce in any circumstance. Once again, because the goal of the annotations was to determine how effective the language models are at producing instructions that can be followed, having instructions that are impossible is vital.

Example of redundant ingredient annotation (from char rnn):
"1 tb baking powder
1/2 ts baking powder
1 ts baking powder"

Example of impossible sequence annotation (from GPT-2):
"Keep hot over low heat until ready to serve. This will chill the soup."

Example of impossible instruction annotation (from word rnn):
"turn into an saucepan."

The method for recording the annotations was this. Annotators would count the number of each error in a recipe and record the number in a spreadsheet. A recipe was defined as any text between two "[END]" characters that the researchers inserted into the training data to be generated by the language model. The researchers chose to record the number errors per recipe instead of recording the area in which errors occurred because the primary interest of the research was the rate at which different models produced errors and because labeling specific locations would significantly slow the annotation process. Recipes were determined as being between "[END]" characters so that the model would be the final determiner of how long recipes would be, and so that the knowledge of the location of "[END]" characters could then be used to interpolate the rate at which the models errored per word, character, and newline.

9 Intrinsic Evaluations

Our intrinsic evaluation of choice for the project was perplexity which was calculated by predicting the next character or word following a sequence. The word-rnn and GPT-2 model predicted sequences of equal length. In addition, Accuracy was calculated for the char-rnn and word-rnn.

Model	**Accuracy**	**Perplexity**
Char-rnn	75.66%	2.297
Word-rnn	36.64%	38.04
GPT-2 fine tuned	N/A	6.404
GPT-2 pretrained, untuned	N/A	5.264

Figure 4: Intrinsic Evaluations on Test Split

The results show that the character level language model was able to achieve the highest accuracy of 75.66% and the lowest perplexity of 2.297. A simple explanation for this is that there are simply fewer choices to pick from between choosing from 125 possible characters versus choosing from thousands of words or thousands of tokens which is what the word-rnn and GPT-2 models do. However, the relatively high accuracy of 75.66% compared to the word-rnn's 36.4% accuracy suggests that the model is doing relatively better at this prediction task. Another likely reason that the character model outperformed the other models, is because predicting a single character can often be an easier task than predicting a word to follow a sequence. The reason for this is that the character may come in the middle or end of the word, which is a context where the number of likely characters would be much narrower than guessing the word it is a part of in the first place. A fairer evaluation may have been to have the language model only guess characters that are the first letter in a word or not a part of any word like a formatting character.

As expected, the fine-tuned GPT-2 model significantly outperformed the word-rnn according to our intrinsic evaluations. Interestingly, the fresh out of the box, untuned GPT-2 model also outperforms the word-rnn in predicting sequences in recipes according to the perplexity score. While it is surprising a model not designed specifically to build recipes would outperform the word-rnn, the GPT-2 model may have seen at least some recipes in its training, and

it has been shown that un-fine-tuned GPT-2 models can be well equipped for some tasks [6].

The last interesting result from the intrinsic evaluations is that the GPT-2 model that had been fine-tuned on the training recipes was narrowly outperformed by the GPT-2 model that had not seen fine-tuning on recipes. This demonstrates that the GPT-2 model tuned for this project must have seen some overtraining on the training data, and further demonstrates the effectiveness of the raw GPT-2 model at this type of word predicting task.

10 Extrinsic Evaluations

This project also evaluated the models with an extrinsic method, utilizing human annotations performed on recipes generated by the models. The raw counts of errors counted by the humans are recorded here.

Model	Redundant Ingredients	Impossible Sequences	Impossible Instructions
Char-rnn	551	64	58
Word-rnn	23	132	197
GPT-2 finetuned	4	14	5

Figure 5: Number of Annotated Errors

Model	Recipes Annotated	Chars	Words	Lines
Char-rnn	30	30245	5677	950
Word-rnn	30	84640	14524	1656
GPT-2 finetuned	30	71112	14265	1569

Figure 6: Annotation Statistics

A more detailed summary of the annotation results for the GPT-2 model is listed below. The same data for the word-rnn and char-rnn will be attached in the appendix.

GPT-2	Redundant Ingredient	Impossible Sequence	Impossible Instruction
Per Recipe (mean)	0.133	0.467	0.167
Per Recipe (median)	0	0	0
Standard Deviation	0.434	0.776	0.379
Per Character	0.0001322	0.0004628	0.0001653
Per Word	0.0007045	0.0024660	0.0008807
Per Line	0.0042105	0.0147368	0.0052631

Figure 7: GPT-2 Annotation Data

The most important takeaways from the detailed analysis of the form of the annotation data are these. The standard deviation was low for all error accounts of the GPT-2 model, but there was more variety in the errors of the other model. In particular, the standard deviation of redundant ingredients found in the char-rnn model was very high at 23.2. The high variance was a result of one of the weaknesses of the char-rnn model. Because the context window in characters is so short, oftentimes the model might continue listing ingredients for multiple pages before randomly deciding to leave the ingredient loop. The context window for the char-rnn was ultimately too short for the model to learn the important syntax of ingredient list as we will discuss in our analysis of the annotation error medians shortly.

Another important observation about the annotations is that the relationship between errors per recipe, per character, per word, and per line remain relatively constant, so for the sake of dealing with the large outliers that exist in the annotations as noted by the high standard deviations for certain categories of data, the comparison of error will be done using the median error per recipe.

When the different categories of error are grouped per recipe, the resulting median values reveal interesting trends. As described in terms of the standard deviation of redundant ingredient with the char-rnn model, The char-rnn model was much more likely to produce redundant ingredients, with a mean number of 6, when compared to the other models with the next largest median number of redundant ingredients being 0.5. The char-rnn appeared to be unable to remember what ingredients it had listed and as a result could go on listing ingredients for an extremely long time. One recipe had 95 repeat ingredients! We believe this is a result of the char-rnn not having a wide enough context window to realize what ingredients it has already mentioned, which is a big weakness in recipes when remembering what ingredients are being employed in a recipe is a big deal.

While the word-rnn may have erred fewer times on redundant ingredients, it had the highest score in producing impossible instructions. The word-rnn model produced many instructions that were incomprehensible with the highest median score of 4. We believe this is because of the large number of labels, which was equivalent to the number of unique words that appeared in the recipes. Although we optimized the labels to exclude scarce words (those appearing less than 5 times in all the recipes), there were still over 15,000 possible outputs. This issue significantly slowed the convergence of the model, affecting how well it could be trained. Although the word-rnn produced more unreadable content than the char-rnn, its generations were generally more varied, using a wider range of ingredients and recipe formatting.

The most important result of the extrinsic evaluations is that the GPT-2 model, as hypothesized, far out-predicted the recurrent models in the extrinsic scoring. The median scores of 0 redundant ingredients, 0 impossible sequences, and 0 impossible instructions is extremely impressive, but maybe a bit misleading. It was common for a recipe to exhibit at least one of the possible errors. The low deviation indicates that there were not many outliers in the data which might also indicate that an evaluation of the mean errors might be more applicable for this specific model. With that in mind, the worst mean score for the GPT-2 model was the impossible sequence error of 0.467. While this score is still higher compared to the recurrent models, it is interesting that this is not the worst scoring category for either of the recurrent models. In fact, each model in this study has a different worst category of extrinsic evaluation. The char-rnn's worst category is redundant ingredient, the word-rnn's worst category is impossible instruction, and the GPT-2 model's worst category is impossible sequence. This result indicates that while the GPT-2 model might be better across the board, every model has unique weaknesses.

Model	Redundant Ingredients	Impossible Sequences	Impossible Instructions
Char-rnn	6	2	2
Word-rnn	0.5	3	4
GPT-2	0	0	0

Figure 8: Average Number of Errors per Recipe

11 Crossover Analysis

While the un-fine-tuned GPT-2 model may have outperformed the fine-tuned model on the intrinsic evaluation, it would fail miserably at generating recipes. This could be verified by the impossibility of the task of performing the annotations described in this project on text generated by un-tuned GPT-2. One observation of the research is that the extrinsic annotating method would be more robust if the process could be performed on a corpus of text that is not the target of annotation. This might allow the analysis of models not designed for the same task to be compared with extrinsic methods. While the value of applying recipe annotations to a corpus of non-recipes might not reveal any insights, it is our belief that the methods presented in this project for extrinsic evaluation might be adapted to different tasks or some other general task. For example, the annotation of impossible sequences might be adapted more generally to identify models that produce sentences that form non-sequiturs.

12 Error Analysis

One obvious error in the study is that the GPT-2 appears to have become overfit on the training data. This may have been possibly avoided with a larger set of training recipes, or with fewer training iterations paying closer attention to the perplexity on the development split.

Another area of error was the perplexity. A few issues exist with our method of intrinsic evaluation. While the word-rnn and GPT-2 models saw nearly identical tasks in predicting the next word in a sequence, the character model did not see an analogous task in predicting the next char. One correction may have been made by having the character model only predict the first characters in words, or even to predict all of the characters in a word. Because the prediction tasks for every model were of the same length sequence, perplexity was not normalized on the lengths of the sequences. It is possible that the internal tokenization of the models might have lent an advantage in un-normalized perplexity.

While many efforts were made to remove bias from the extrinsic annotations, there are a few sources for possible error. 30 recipes were annotated with each researcher completing a similar number of recipes from each model. This was done to mitigate biases inherent in different annotator's judgements of error. 30 recipes was a significant amount of recipes seeing the annotators read thousands of lines of text, but with more annotators and more annotated recipes more precise results might be achieved. A significant source for error may have been the lengths of the recipes. While we were careful to record the rate of error per line, recipe, word, and character, different parts of recipes can have different lengths. For example, the char-rnn obviously produced more lines of ingredients than of instructions, which biased it towards having higher rates of redundant ingredients. A study controlling for the ratio of ingredient to instruction might see different results. However, it is our belief that the large amount of ingredient lists produced is an inherent flaw of the char-rnn model and it was our studies intent to measure this. A final reflection on the annotation process is that the GPT-2 text was simply easier to read. This meant that annotations could be performed more quickly on this model. The fact that the text generated by the other models was harder to read, means that it may have been easier to miss sentences that should have been marked as errors, because it can become easy to skip over nonsense while reading when there is so much of it in one place.

13 Work Division

Both of us worked on annotating the models' generations. Will handled the majority of the word and char rnn implementations, tuning, and testing. Alexander set up the GPT-2 model and found the models' perplexities.

14 Conclusion

The results of our experiments clearly highlight the advantages of GPT-2 over conventional models in generating artificial food recipes. From annotating, it is evident that LSTM-based models are incapable of capturing the nuance needed in making a cohesive, logical recipe, even if the model appears to have a high intrinsic ability.

We found that the GPT-2 model on average produced recipes that are more readable and have more permanence than that of our sources [2, 3]. Its generated recipes appear to follow logical steps, instead of reading like several disjoint instructions.

It is an observation of this study that the task of generating ingredient lists is not the same as the task of generating instructions. This is reflected in the choice of having errors specific to each of these sub tasks of recipe generation. An extension of this project might be to make a model designed to execute one or the other of the sub tasks and identify if it can perform these tasks better, extrinsically, when not trying to learn both at the same time. It is likely that multiple language models suited to these subtasks working in concert might perform better at creating human friendly recipes.

References

Richard Brant. “Chef Watson has arrived and is ready to help you cook.” IBM Blogs, 2016.

Tom Brewe. “Do androids dream of cooking? : char-rnn recipes.” github, 2017.

Stephen Johnson. “This Neural Network Is Generated Some Truly Bizarre Recipes.” Big Think, 2017.

S.F. Svistova. "ANALYZING COOKING RECIPES WITH MACHINE LEARNING". Meždunarodnyj naučno-issledovatel’skij žurnal (International Research Journal) № 08 (74), (2018): 44. Tue. 21. Aug. 2018.

Amaia Salvador, Michal Drozdzal, Xavier Giro-i-Nieto, Adriana Romero. “Inverse Cooking: Recipe Generation from Food Images.” arXiv:1812.06164v2 [cs.CV], 15 Jun 2019.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. “Language Models are Unsupervised Multitask Learners.” OpenAI, 2019.

A Appendix – RNN Annotation Data

Char-rnn	Redundant Ingredients	Impossible Sequences	Impossible Instructions
Per Recipe (mean)	18.366	2.133	1.933
Per Recipe (median)	6	2	2
Standard Deviation	23.260	1.479	1.574
Per Character	0.006509	0.000756	0.000685
Per Word	0.037937	0.004406	0.003993
Per Line	0.332729	0.038647	0.035024

Word-rnn	Redundant Ingredients	Impossible Sequences	Impossible Instructions
Per Recipe (mean)	0.766	4.4	6.566
Per Recipe (media)	0.5	3	4
Standard Deviation	1.072	4.568	7.749
Per Character	0.000323	0.001856	0.002770
Per Word	0.001612	0.009253	0.013810
Per Line	0.014659	0.084130	0.125557