Cooking with BERT
Project Proposal

Alexander Schott, William Braga

Abstract

Computer generated food recipes have been an area of research interest as early as 2016 when IBM published a cookbook with chef Watson. 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, but impossible instructions for a human to execute. In this project we will evaluate text generating models, with an extrinsic measure of plausibility, by using human annotation of computer-generated recipes. Additionally, we will evaluate the efficacy of transfer learning to the food recipe task using a model developed from the pretrained BERT model.

1 Introduction

There currently exist recipe generators and other food language projects that operate under a variety of different models, including character-level recurrent neural networks, word2vec, and long short-term memory networks. At first glance, these generators can produce results that appear legitimate. An LSTM may give helpful advice - “insert a knife or a metal skewer into the cake and if it comes out clean, then the cake is ready to devour in” – and a word2vec may create the analogy, “egg is to bacon as orange juice is to coffee” [3, 8]. However, their shortcomings become all too apparent upon a closer look. In the same recipe, the LSTM distinctly notes “vanilla essence is important to camouflage the smell of eggs” [8]. If one were to trust a char-rnn recipe, they may end up serving the raw combination of milk, sugar, vanilla, seasoned flour, water, cornstarch, tomatoes, oregano, and nutmeg [1].

Although current models are effective at finding and joining similar ingredients and instructions, they are inadequate at processing the validity of the recipe as a whole. For example, a model should not produce a recipe titled “BEEF (MARINADE)” and not mention any meat products in the ingredients list or directions [1]. In addition, several recipes fall victim to logistic impossibilities, like saying to cook and cool ingredients simultaneously or to use cooking devices in ways that are infeasible.

Our proposal involves creating and training a model, namely BERT, that will outperform existing generators. We will also make simpler models, char-rnn and word-rnn, as bases for comparison. The inclusion of an extrinsic evaluation measure will allow us to rate the recipes beyond their perplexity, or predictive ability. We expect BERT to showcase a level of depth and understanding beyond that of a more basic rnn. Through our annotations, we will be able to capture nuanced differences in a way intrinsic evaluation cannot.

Once a comparison of these models has been completed, we hope it may serve as a reference tool for emerging projects in recipe generation. If our BERT model can create recipes with more effective ingredient permanence and instruction comprehension, BERT may become the new baseline for related tasks. With artificial recipes that are sensical and reasonable, a wide array of practical use cases will open, like intelligent meal recommendations or smart ingredient substitution.

2 Implementation

In order to learn the text generating models we will use the dataset of recipes available at the Now Your

Cooking! Recipe Software website. The website has a formatted data-set of over 150K recipes, and there are over 500K more available in various other data-sets on the website. This database has been used for similar tasks in the past [1].

We plan to produce three recipe generating models. The first, is a character level rnn. The char-rnn is a common approach to generating recipes and is expected to exhibit common flaws in producing steps that are impossible [1]. The second model will be a word level rnn with word embeddings [2, 3]. This model is expected to perform better, because it should be easier for the model to fit more words in its context window at the same time, and recipe steps often rely on information that occurred in a step long before it. For our third model we plan to utilize transfer learning using the BERT model [2]. This model is expected to perform the best, because it will be able utilize knowledge about language learned outside the scope of the dataset leveraged for the other models.

Another important step to this project is extrinsic evaluation of model performance. Where intrinsic measurements like perplexity and loss do well to describe how well given phrases are fit to the model, they do not give a simple answer to whether an action described in the text is possible. By annotating computer generated samples for instances of impossible instructions, a frequency can be calculated for how often the model is producing logical errors. We will label sentences in which an impossible instruction occurs. We will use this labeled data to identify the average number of impossible instructions per recipe as well as the number of words generated by the model per impossible instructions generated by the model.

Because recipes have a finite number of instructions, it is important the model also be able to decide when the recipe will end. In order for the model to learn where recipe ends occur in order to reproduce the phenomena, a recipe end tag will be inserted at the end of every recipe before training.

Should something go wrong in the project and a pivot to a backup plan be needed, we should be able to achieve this simply by limiting the scope of the project. The most likely areas for failure would be in the BERT model and the extrinsic evaluation. Should one of these plans not pan out, we should be able to fall back on our other models and intrinsic evaluations for our project. Also, if BERT were to fail, we could attempt to replace it with a different LSTM model.

3 Plan

Project Timeline:

Week 1: Format and extract information from the data-set.

Week 2: Implement and test the char and word-rnn. Conduct midterm report.

Week 3: Implement and test the BERT model.

Week 4: Finish tuning the hyperparameters for all models.

Week 5: Perform extrinsic evaluation through annotations.

Week 6: Analyze and compare evaluations and conduct final report.

Dividing the labor fairly between teammates ought to be mostly straightforward. We have two teammates and are planning on building three models and doing a significant amount of hand labeling. As long as each teammate produces at least one model on their own and is responsible for tuning it and helping with other coding challenges, this is thought by the team to be a reasonably fair split. The teammates expect their partners to complete a roughly equal amount of annotations and to work on the written assignment and presentation together. In this way everyone will work on a little bit of everything and complete roughly equivalent amounts of work.

At the end of each week, we plan to discuss how much progress we made on our work. Depending on how far along we are, we may decide to reshape the schedule or allocate different amounts of time per task. Our focus is to have all our models completed and trained and our evaluations compared by the six-week mark. By meeting this deadline, we expect to have all the information necessary to write the final report. Due to unforeseen circumstances, the division of labor may be temporarily adjusted away from one team member with the expectation that he will pick up more slack later on or on a different task.

References

[1] Tom Brewe, “do android’s dream of cooking.” github, 2018. https://gist.github.com/nylki/1efbaa36635956d35bcc

[2] Wesley Tansey, Edward W. Lowe Jr, James G. Scott, “Diet2Vec: Multi-scale analysis of massive dietary data.” NIPS Workshop Machine Learning and Health, 2016.

https://arxiv.org/abs/1612.00388

[3] Jaan Altosaar. “food2vec - Augmented cooking with machine intelligence.” 2018. https://jaan.io/food2vec-augmented-cooking-machine-intelligence/

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” 2018. https://arxiv.org/abs/1810.04805

[5] Richard Brandt, “Chef Watson has arrived and is ready to help you cook.” ibm blogs. 2016. https://www.ibm.com/blogs/watson/2016/01/chef-watson-has-arrived-and-is-ready-to-help-you-cook/

[6] Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, Julian McAuley. “Generating Personalized Recipes from Historical User Preferences.” https://cseweb.ucsd.edu/~jmcauley/pdfs/emnlp19c.pdf

[7] Amaia Salvado, Michal Drozdzal, Xavier Giro-i-Nieto, Adriana Romero. “Inverse Cooking: Recipe Generation from Food Images.” 2019. https://arxiv.org/pdf/1812.06164.pdf

[8] Neha Setia. “Generate recipes using multi-ingredient aware LSTM.” 2019. https://developer.ibm.com/technologies/artificial-intelligence/tutorials/develop-a-multi-topic-aware-long-short-term-memory-mta-lstm-network/