Abstract

This paper investigated the use of deep learning models for recipe generation from image inputs. The authors investigated the model architecture presented in [reference paper], looking at performance over a smaller dataset, investigated the effects of various introduced parameters (learning rate, training mismatch), and identified methods of potential architecture improvement (transformers). Ultimately, the authors found that insufficient data size and training time resulted in poorer performance relative to the benchmark, however the investigated alternative architectures appear promising avenues for further research.

1. Introduction/Background/Motivation

This paper investigated the use of deep learning models for recipe generation. Specifically, it looked at how a deep learning model can take a food image, parse the underlying components, and produce a set of outputs (ingredients, recipes) useful to an end user (im2recipe); a process that can also be done in reverse (recipe2im).

This is an interesting problem as it shows applications of deep learning models for practical use; the recreation of a food one saw online, a calorie counter for one on a diet, or understanding of food ingredients for allergy considerations.

Specifically, we’re interested in learning how to build these models, understanding and potentially improving on the used architectures. Our dataset [1] contains cross-modal data (both image and textual info). As such, our approach enables us to investigate both CNN and LSTM architectures - an exciting prospect as it melds multiple aspects of the course together.

The current state of the art [1] uses a dual CNN and LSTM architecture. The CNN is pretrained on a ResNet 50 architecture with a primary approach of freezing the model and training the terminal layer. Two LSTMs are used; a bi-directional LSTM to learn the ingredient list (used since order is unstructured) and a uni-directional LSTM to learn the recipe instructions. Additional details are discussed throughout this paper for comparison. [add in pertinent metrics for comparison?]

1. Approach

2.1. Our approach

The first task was to generate the architecture presented in [1] solely from the presentation in the paper. From this, we wanted to see how one might recreate the original author’s work just from their published instructions. While the author’s instructions were quite detailed, after analyzing their dataset and training process, it was immediately evident that we would be unable to recreate the entire process because the dataset was too large (over 1M images) and we had limited compute capabilities and time. Therefore, we took as a base assumption that we would only investigate the model architecture, utilizing the source data and pre-processing provided by the authors in [reference link to github page].

We next looked how our results model architecture results compared to those of the original authors. We tried a few different approaches as discussed further in section 3. On the CNN, we investigated both freezing and unfreezing the pretrained layer weights. On the LSTM, we looked at a basic pytorch LSTM implementation, pytorch LSTMs with sequence packing (a commonly used technique [reference to pytorch documentation] to help train the model faster using fewer computations), and tried to replace the LSTMs entirely with transformer encoders. Of note, the transformer used was a custom built transformer used for CS7643 Assignment 4, modified for purposes here.

Additionally, for each of the above implementations, we tuned parameters to try to find the optimal results. Discussed more in Section 3, we looked at loss as the primary metric, as this was most telling to performance observed. To compare results to the original paper, we also looked at median rank (the ranking of model results compared to a random sample of 1000 other potential results), and we looked at accuracy results as there were more stable in our training experiments compared the to the median rank metric.

[go into more detail about training here or in experiments? Or in section 4?]

Discuss im🡪recipe output?

2.2. Problems

The first anticipated problem related

2.3. Code repos used and changes made.

Use of paper

Used paper to define architecture, we created architecture from scratch using pytorch methods (e.g. nn.cnn, nn.lstm, etc)

1. Things to discuss
   1. Dataset
      1. b/c of size (TBs of data), we only used the testing set. Assuming test set well represents underlying data (otherwise wouldn’t be accurate in paper tests)
   2. Tuning
   3. Semantic regularization?
   4. Transformers?
2. Work done
   1. Justin
      1. developed the main.py function to run the code for necessary use in this project
      2. modified data\_loader to load appropriate data for both image and text analysis
      3. generated and tuned the CNN architecture for image processing in accordance with <cite paper>
   2. Geoff
      1. Set up and populated initial github (for code base), box (for data store), and jupyterlab repository code & interfaces to enable collaboration and to utilize cloud services
      2. Wrote grid search training script and plotting methods to help tune parameters and display results
      3. generated and tuned the LSTM architecture for ingredient and image processing in accordance with <cite paper>
3. Database

Recipe 1M+ [1]: <http://im2recipe.csail.mit.edu/>

1. References

[1] J. Marín et al., "Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 187-203, 1 Jan. 2021, doi: 10.1109/TPAMI.2019.2927476.

[2] J.-J. Chen, C.-W. Ngo, F.-L. Feng, and T.-S. Chua, “Deep understanding of cooking procedure for cross-modal recipe retrieval,” in Proc. 26th ACM Int. Conf. Multimedia, 2018, pp. 1020–1028. [Online]. Available: <http://doi.acm.org/10.1145/3240508.3240627>

[3] M. Carvalho, R. Cadene, D. Picard, L. Soulier, N. Thome, and M. Cord, “Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings,” in Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2018, pp. 35–44.

[4] M. Kumari and T. Singh, "Food Image to Cooking Instructions Conversion Through Compressed Embeddings Using Deep Learning," 2019 IEEE 35th International Conference on Data Engineering Workshops (ICDEW), 2019, pp. 81-84, doi: 10.1109/ICDEW.2019.00-31.

[5] H. Rawlani, J. Saita, V. Zambre and R. L. Priya, "Deep Learning based approach to suggest recipes," 2018 International Conference on Smart City and Emerging Technology (ICSCET), 2018, pp. 1-4, doi: 10.1109/ICSCET.2018.8537350.