Robots in the Kitchen: NeuralNourishment

James Tanner, Paul Vander Vort, Aidan Weingrad

Introduction

If you have ever visited the *Lego House* in Aarhus, Denmark, then you may be familiar with the thought of robots in the kitchen. Their playful take on automated culinary creation sparked a journey into a particularly niche area of kitchen research. In our project “Neural Nourishment,” we blend the art of cooking with the mathematical precision of artificial intelligence. This initiative leverages deep learning to innovate the way we create recipes, making cooking both easier, faster, and more enjoyable.

Lego House in Aarhus, Denmark

Background on NeuralNourishment

While AI has been involved in recipe generation for some time, recent advances have boosted its potential to change cooking practices significantly. Pioneering projects like Generative recipe & ChatGPT powered nutrition assistance for sustainable cooking and Ratatouille: A tool for Novel Recipe Generation gave us inspiration for both general ideas and technical approaches to our project. To provide a fresh take on this idea, we chose to create a model that could work with just the ingredients a user has already. The goal being to reduce food waste, provide novel meals, and promote creativity in the kitchen (with a bit of help). We leverage the extensive RecipeNLG dataset with over 2.2 million cooking recipes, training our model using TensorFlow and the KerasNLP library, along with a pretrained GPT model.

Ratatouille

Technical Overview

- Dataset Setup: Our dataset comprises strings of complete recipes, adjusted from the CSV provided Kaggle dataset to a JSON format for easy processing. The RecipeNLG, sourced from Kaggle, structures the ingredients into plain and generalized ingredient names. For example, “packed brown sugar” and “dark brown sugar” would both be translated to “brown sugar”. This data cleaning allows our neural network to learn from a structured representation of each recipe’s more obvious components. For our purposes, we utilized the “NER” and “directions” sections of the dataset.

title	ingredients	directions	link	source	NER
1312871 unique values	2226362 unique values	2211644 unique values	2231142 unique values	Gathered 74% Recipes1M 26%	2133496 unique values
No-Bake Nut Cookies	["1 c. firmly packed brown sugar", "1/2 c. evaporated milk", "1/2 tsp. vanilla", "1/2 c. broken nuts...	["In a heavy 2-quart saucepan, mix brown sugar, nuts, evaporated milk and butter or margarine.", "St...	www.cookbooks.com/Recipe-Details.aspx?id=44874	Gathered	["brown sugar", "milk", "vanilla", "nuts", "butter", "bite size shredded rice biscuits"]
Jewell Ball'S Chicken	["1 small jar chipped beef, cut up", "4 boned chicken breasts", "1 can cream of mushroom soup", "1 c...	["Place chipped beef on bottom of baking dish.", "Place chicken on top of beef.", "Mix soup and crea...	www.cookbooks.com/Recipe-Details.aspx?id=699419	Gathered	["beef", "chicken breasts", "cream of mushroom soup", "sour cream"]
Creamy Corn	["2 (16 oz.) pkg. frozen corn", "1 (8 oz.) pkg. cream cheese, cubed", "1/3 c. butter, cubed", "1/2 t...	["In a slow cooker, combine all ingredients. Cover and cook on low for 4 hours or until heated throu...	www.cookbooks.com/Recipe-Details.aspx?id=10570	Gathered	["frozen corn", "cream cheese", "butter", "garlic powder", "salt", "pepper"]
Chicken Funny	["1 large whole chicken", "2 (10 1/2 oz.) cans chicken gravy", "1 (10 1/2 oz.) can cream of mushroom...	["Boil and debone chicken.", "Put bite size pieces in average size square casserole dish.", "Pour gr...	www.cookbooks.com/Recipe-Details.aspx?id=897570	Gathered	["chicken", "chicken gravy", "cream of mushroom soup", "shredded cheese"]

From https://www.kaggle.com/datasets/paultimothymooney/recipenlg

- Model Architecture: We utilize a pretrained GPT-2 model developed by OpenAI, a type of Transformer that is particularly effective for generating text. This model’s architecture enables it to consider a full sequence of words at once, which is ideal for understanding and generating recipes. This decision was also weighed against using an LSTM, but we decided against this due to a transformers more optimized training method on our GPUs, as well as limitations in complexity for LSTMs.
- Training: This model has the capability of optimized training, utilizing Google tensor processing units (TPUs). With the help of TPUs, we can train our model efficiently, processing data in large batches and speeding up a learning process that is fairly taxing on our Apple M1 chips.

```
# train on TPU if appropriate
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
    print('Running on TPU ', tpu.master())
except ValueError:
    tpu = None

if tpu:
    tf.config.experimental_connect_to_cluster(tpu)
    tf.tpu.experimental.initialize_tpu_system(tpu)
    strategy = tf.distribute.experimental.TPUStrategy(tpu)
else:
    strategy = tf.distribute.get_strategy()

print("REPLICAS: ", strategy.num_replicas_in_sync)
print("GPUS: ", tf.config.list_physical_devices('GPU'))


# Check GPU availability
print("GPU Available:", tf.test.is_gpu_available())

# Check TPU availability
tpu_available = False
devices = tf.config.list_logical_devices()
for device in devices:
    if device.device_type == 'TPU':
        tpu_available = True
        break

print("TPU Available:", tpu_available)
```

Transformer Model Explanation

At the core of our project lies the Transformer model used by GPT-2, renowned for its innovative use of attention mechanisms. These mechanisms enable the model to focus dynamically on different segments of the input data. Unlike previous models that processed text linearly, Transformers consider the entire sequence simultaneously, allowing for a richer understanding of context. This feature is vital for recipe generation, where the interaction between ingredients and their sequence can significantly impact the coherence and plausibility of the resulting recipes. The self-attention mechanism, a hallmark of the Transformer, ensures that every word—or in the case of our project, every ingredient or cooking instruction—is evaluated in the context of every other word in a recipe. This global perspective helps the model maintain a

focus on the overall dish being prepared, rather than just the immediate step, ensuring that each generated recipe is both grammatically sound and culinarily viable.

Learn more about transformer models in this great medium.com read!

Implementation Details

Our implementation begins with the crucial step of tokenization, where we employ WordPiece Tokenization. This approach breaks down the text into smaller, manageable pieces, which the model can process more effectively. By optimizing tokenization, we ensure that the model learns from a clean, orderly dataset, which enhances both the efficiency of the training process and the quality of the text generation. This is particularly important in the culinary domain, where the diversity of ingredients and cooking terms can vary greatly.

Tokenize the dataset

We train a WordPiece tokenizer on the dataset, reserving special tokens for the beginning and end of recipes. We can load the vocabulary and use the Keras `WordPieceTokenizer` to tokenize our tensors within the `tf.data` pipeline.

In [ ]:
```
# train the tokenizer's vocabulary
vocab = keras_nlp.tokenizers.compute_word_piece_vocabulary(
    data=dataset,
    vocabulary_size=VOCAB_SIZE,
    reserved_tokens=SPECIAL_TOKENS,
)

# save the vocabulary (so this step can be skipped in the future)
with open(VOCAB_FILE, 'wb') as f:
    pickle.dump(vocab, f)
```

In [ ]:
```
# load the vocabulary
with open(VOCAB_FILE, "rb") as f:
    vocab = pickle.load(f)

# load the tokenizer object with the trained vocbulary
tokenizer = keras_nlp.tokenizers.WordPieceTokenizer(
    vocabulary=vocab,
    sequence_length=SEQ_LEN,
    special_tokens_in_strings=True,
    special_tokens=SPECIAL_TOKENS,
    oov_token=OOV,
)
```

Following tokenization, we fine-tune the pre-trained GPT-2 model on our specially curated dataset, the RecipeNLG. This dataset offers a rich variety of cooking recipes, providing the diverse culinary context needed for the model to adapt its capabilities from general language understanding to specific, recipe-oriented tasks. Fine-tuning is a critical phase where the model's pre-existing knowledge of language, gained from extensive pre-training on a broad range of texts, is honed to focus on culinary text. This process allows the model to transfer its expansive linguistic capabilities to the more specialized domain of recipe generation, enabling it to produce novel and creative recipes that go beyond standard or traditional dishes, reflecting a genuine understanding of culinary concepts.

Finetune the model

In [ ]:

```
learning_rate = keras.optimizers.schedules.PolynomialDecay(
    5e-5,
    decay_steps=dataset.cardinality() * EPOCHS,
    end_learning_rate=0.0,
)
loss = keras.losses.SparseCategoricalCrossentropy(from_logits=True)

gpt2_lm.compile(
    optimizer=keras.optimizers.Adam(learning_rate),
    loss=loss,
    weighted_metrics=["accuracy"],
)

checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath='checkpoints/transfer_learning_cp_{epoch:02d}.keras',
    save_best_only=False,
)
text_generation_callback = TextGenerator(k=10)

callbacks = [
    checkpoint_callback,
    text_generation_callback,
]

gpt2_lm.fit(
    dataset,
    epochs=EPOCHS,
    callbacks=callbacks,
)
```

Results and Analysis

As for the tasty results, this model is generating novel and unique recipes that we can't wait to try. The output is structures as the directions section of a recipe, similar to that found in the RecipeNLG dataset. We'll be trying out some of the outputs of our model in the test kitchen. We're excited to see if the patterns that our model found between recipe pairings translates to a real-life culinary experience.

As far as user interaction is concerned, so far we are working with a simple interface that includes an ingredient list, with plans of adding a dietary restrictions section as well.

Challenges and Lessons Learned

During the project, we faced challenges related to computational demands and data quality. These challenges taught us valuable lessons in managing large datasets and refining our model to improve performance on our limited hardware. Additionally, selecting the proper model architecture proved to be a significant task. While some tasks are fairly straightforward, the abstract idea of "what makes a recipe good?" can be hard to conceptualize. This translates to difficulty finding the right architecture to search for the right types of patterns in our data.

Future Directions

As for the future of NeuralNourishment, we feel it has a bright path ahead. The nine to five worker will always need to eat at the end of the day. This alternative to ordered and premade food has potential to reignite a fervor for cooking and creativity. The hope is to place a tool like

this into the hands of everyone, from aspiring chefs to people trying to feed their kids in a healthy and sustainable way.

References

https://github.com/jamestannner/NeuralNourishment

RecipeNLG (cooking recipes dataset)
*2,231,142 cooking recipes (.csv)*www.kaggle.com

"Attention is All You Need", Vaswani et al.

"Language Models are Few-Shot Listeners", Brown et al.