Recipe Infusion: Personality-Infused Recipe Generation using Text Style Transfer

Quoc Anh Bui
qhbui@umass.edu

Anshumaan Chauhan
achauhan@umass.edu

Ashvath Balgovind
abalgovind@umass.edu

Saranath Kannan
saranathkann@umass.edu

1 Introduction

Text is a mix of content (information) and lexical and syntactical attributes. The style of a text is determined by these syntactical attributes. Text style captures many different features in a passage. Formality, sentiment, and vocabulary are just some of the characteristics that contribute to a text's style and form the differences between authors and various works. The goal of Text Style Transfer (TST) is to convert the original style of a given text to a target style while preserving the original content. TST aims to do this by relying on two attributes (Jin et al., 2022): content of the original sentence, and the target style. This is different from style conditioned language modeling, which is only conditioned on a single style attribute. Applications of text style transfer include intelligent bots and writing assistants, text paraphrasing, and smart content moderation.

Recipe generation is a semi supervised text generation task that takes ingredients as input and outputs complete recipes using the ingredients provided (Goel et al., 2022). There are several components that need to be considered to form a recipe such as ingredients, quantity, utensils, and steps (along with the order of execution). In general, data used to train a recipe generation algorithm consists of structured pieces of text, which can be used in a supervised learning approach to train a Language Model (LM) to generate new recipes. Therefore, it is feasible to fine-tune a decoder-only transformer model such as GPT-2 (Radford et al., 2019), Transformer XL (Dai et al., 2019) or CTRL (Keskar et al., 2019), which can be accessed through the Hugging Face Transformer library (Wolf et al., 2020). We use these decoder-only architectures to generate recipes, and will infuse the content of these recipes with styles of various celebrities, authors, and musicians via TST.

The main issue in TST (a sub-field of NLP) is the lack of matched input and output sentences that have the same content, but use different styles (parallel data). To overcome this hurdle, three Non-Parallel data approaches are available - Disentanglement, Prototype Editing and Pseudo Parallel Corpus Generation. We will be using a Generative Pseudo Parallel Corpus Generation approach known as Iterative Back Translation (IBT) to create our style translation dataset. Inspired from (Iyyer et al., 2020)(Syed et al., 2020) we will task an Encoder-Decoder architecture based Transformer to learn the style of different personalities (one for each personality) (Fu et al., 2017) (Prabhumoye et al., 2018) (Lin et al., 2020) using synthetic supervised data created using Back Translation. After training, the Encoder-Decoder architecture will take a base recipe as input, and will output multiple recipes that have the same content and instructions as the base recipe, but are flavored with their own distinct styles.

In this paper, we are presenting a framework called '*Recipe Infusion*', which generates persona-infused recipes using a combination of 2 simple Transformer architectures. Our main contributions in this paper are:

- Construction of a rich, synthetic supervised dataset capturing linguistic attributes of several personalities.
- Simple framework utilising a pair of pre-trained transformers for style transfer of a generated recipe in an unsupervised manner, while preserving the content.

2 Literature Survey

2.1 Text Style Transfer

Previous research efforts related to Text Style Transfer, including both parallel and non-parallel approaches, are summarized in (Jin et al., 2022) (Hu et al., 2022). Due to the unavailability of a structured dataset, the parallel approaches are only applied to a subset of style attributes such as Formality, Politeness, Gender, Humor and Romance, Biases, Toxicity, Authorship, and Simplicity. The issue with the limited size and availability of parallel datasets was solved using different approaches, such as leveraging the use of external dictionaries containing mappings from Shakespeare's words to Modern English Language words (Jhamtani et al., 2017), and compiling a huge dataset which consists of 33 different Bible versions in English (Carlon et al., 2018). (Pryzant et al., 2018) constructed a neutral style corpus Wiki Neutrality Corpus (WNC) and used it, along with an encoder-decoder architecture model, to neutralize subjective bias from sentences. There is also a cross-alignment methodology for auto-encoders that makes use of two latent representations/distributions. One distribution is obtained during translation from the original style to the target style, whereas the other is learnt when the target style is converted back to the original style representation (Shen et al., 2017). All of the above cases used some form of modified parallel data. Our approach is different because we are compiling our own non-parallel dataset, which will be utilized to train our style transfer transformer.

For non-parallel datasets, (Fu et al., 2017) proposed a neural sequence-to-sequence model consisting of one encoder and multiple decoders. This model is trained in an unsupervised adversarial manner for multi-class style classification (using an encoder and a multilayered perceptron model) and style transfer (utilising a generative decoder). In contrast, (Prabhumoye et al., 2018) implemented back translation to weaken the style embedded in the input text. The latent representation is then used by several decoders to generate text in multiple styles. A combination of both the adversarial and back translation approach was presented in (Lin et al., 2020). The main distinction between the above approaches and ours is that we are using an iterative back translation approach and comparing the performance with approaches such as identity translations and parallel translation.

2.2 Recipe Generation

(Salvador et al., 2019) proposed a sequential encoder-decoder model which takes a food image as input, and outputs its recipe by recognizing ingredients in the image. Ratatouille (Goel et al., 2022) is a GPT-2 (medium) based recipe generation model that is trained on a preprocessed RecipeDB dataset with a BLEU score of 0.806. We approach the same problem with an additional task of styling the recipe generated with several different personas.

3 Data

3.1 Recipe Generation

There are several structured datasets such as Recipe1M+ (Marin et al., 2019), RecipeDB (Batra et al., 2020) and RecipeNLG (Bień et al., 2020). We initially aimed to combine RecipeNLG (built on top of Recipe1M+, and contains additional recipes extracted from web pages) with RecipeDB to have a large dataset for finetuning our LLM. Unfortunately, the complete RecipeDB dataset is not available publicly. A good alternative to RecipeDB, RecipeBox, was used instead. RecipeBox contains recipes from trusted sources such as Food Network, Epicurious, and All Recipes.

3.1.1 Data Preprocessing

The datasets chosen for the task of Recipe Generation vary significantly in terms of how their useful information, such as instructions and ingredients, are formatted. Because we are only concerned with the ingredients and instructions for each recipe, all the other information from the datasets is dropped. The next few preprocessing steps are ordered as follows:

1. Ingredient Processing (RecipeNLG): Ingredients in the RecipeNLG dataset are enclosed within the quotations. For ease of use in the future, as well as to make it compatible and aligned with the ingredient information in the RecipeBox dataset, we remove all the quotations.
2. Special Tokens Removal: Special tokens such as quotes, square brackets, newline characters, etc. are removed to make the data contain only text and basic punctuation such as commas and full stops (or periods).
3. Removal of Incomplete Recipes: We remove all recipes that do not contain ingredients, or

are missing the instructions. This will prevent the model from fine tuning on samples that have missing information.

4. Short and Long Recipe filter: The next step is to remove all recipes that are too short, or too long. We also removed all recipes that were longer than the average recipe length for the dataset. Removal of shorter recipes was threshold based, which can be changed based on user preference. Removal of shorter length recipes was done to prevent the model learning from the recipes which are prone to missing steps in the instructions. Longer recipes were removed to save some computational resources, and to ensure that the recipe length is within the range of maximum input size allowed by the style transfer model.

5. Redundant Recipes Removal: As a precautionary step, to prevent the model from overfitting on redundant examples, we tried to remove recipes that had a cosine similarity score above a certain threshold from the dataset. Due to the enormous size of the dataset, however, the system always crashed. As a result, this step was not executed when creating the final dataset.

6. Tokenization: All the recipes were tokenized using the GPT2 tokenizer, which included additional steps, such as truncation if the recipe length was longer than the model's max input length, and padding if it was shorter.

Some statistics about the number of samples, and average number of words in the recipe of the final dataset in comparison with the original dataset is presented in Table 1. 90% of the final dataset (≈ 579K samples) was utilized in the finetuning process, and the model was later tested on the remaining 10% of the dataset (≈ 64K samples), for BLEU score and Perplexity.

	#Samples		Avg #words	
Dataset	Before	After	Before	After
RecipeNLG	2.2M	577K	98.69	77.52
RecipeDB	125K	65K	230.25	58.65
Final	-	643K	-	85.83

Table 1: Recipe Generation Dataset statistics before and after preprocessing

3.2 Text Style Transfer

For the task of Text Style Transfer, we will be extracting textual content containing the style of the following celebrities/characters: Donald Trump, Taylor Swift, William Shakespeare and Michael Scott, and combine them into a single rich database. Though there is a Kaggle repository that contains the tweets of some famous personalities, there are instances where the model is not able to learn the style properly due to short tweets and an insufficient number of tweets. Therefore, we will be extracting information from different use cases such as song lyrics, captions from YouTube videos, and TV series. Information gathered will be separated based on the celebrity involved and then will be added to a structured database. All of the datasets are non-parallel except for William Shakespeare, which will be utilised to compare the performance of the Encoder-Decoder model on different forms of datasets.

3.2.1 Data Preprocessing

The datasets are collected from different sources and have different formats. Each of them is separately preprocessed to get them into a single common structure that will be used to train our Text Style Transfer model (T5-small). Similar to Recipe Generation, the first step is to delete all the non-useful columns/information from the dataset and only retain the main content - style captured sentences.

Tweets of Donald Trump from until June 2020 were processed by applying a function for removal of URLs and @ mentions with the use of Regular expression library *re*.

The Taylor Swift song lyrics dataset included lyrics for an entire song as each entry. Initially, we grouped (or concatenated) all the sentences for the same song into one big paragraph. But it resulted in a smaller size dataset (≈ 94 songs in total) which lead to poor performance of the Text Style Transfer model. Therefore, we segmented each song down to the sentence level, which gave us more samples to train on.

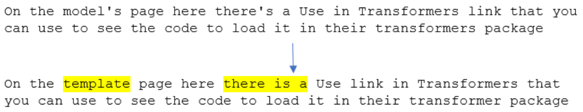
Michael Scott's dialogues were extracted from the script of the show *The Office*. After removal of special characters from the dataset, it is filtered based on each lines's length. We have removed all dialogues that contained fewer than two words (to prevent the model from learning any sort of *one-to-one mapping*).

Lastly, we process the parallel dataset William Shakespeare. It consists of paired, line by line data that matches each Shakespearean line to a Modern English interpretation. The processing is exactly same as Michael Scott's - removal of special characters followed by a length based filtering.

3.2.2 Synthetic Data Generation

Other than our Shakespeare dataset, our other datasets are non-parallel and unsupervised. That is, we do not have paired inputs with our personas' styled text and the un-styled meaning.

Therefore, to create a synthetic/artificial supervised dataset, we perform two different mechanisms: Back Translation and Idenity Translation. Back Translation, as discussed earlier, will remove any style from the input, while preserving the context as much as possible. We converted the English styled text to French, and then translated them back to English. We used two different finetuned MarianMT models from Hugging face for the task of Back Translation - Helsinki-NLP/opus-mt-en-fr (English to French) and Helsinki-NLP/opus-mt-fr-en (French to English). These models use sentence piece as the tokenization strategy which has a limit of 512 tokens on the input. The back translated (style-stripped) text acts as the input sentence to the TST model, and the original English text (style-injected) is the label (Figure 1) - creating a synthetic parallel dataset for supervised training of Text Style Transfer model. In the Identity Translation, the input is also used as the target output.



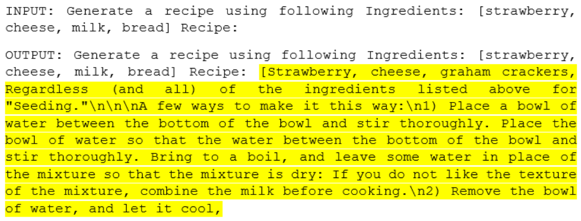
```
On the model's page here there's a Use in Transformers link that you
can use to see the code to load it in their transformers package
```

```
On the template page here there is a Use link in Transformers that
you can use to see the code to load it in their transformer package
```

Figure 1: Back Translation Example

	#Samples		Avg #words	
Dataset	Before	After	Before	After
Trump	41K	41K	20.56	19.43
Taylor	4K	4K	7.24	7.24
Michael	9K	9K	13.78	13.78
Shakespeare	51K	50K	16.82	17.21

Table 2: Text Style Transfer Dataset Statistics before and after preprocessing



```
INPUT: Generate a recipe using following Ingredients: [strawberry,
cheese, milk, bread] Recipe:

OUTPUT: Generate a recipe using following Ingredients: [strawberry,
cheese, milk, bread] Recipe: [Strawberry, cheese, graham crackers,
Regardless (and all) of the ingredients listed above for
"Seeding."\n\n\nA few ways to make it this way:\n1) Place a bowl of
water between the bottom of the bowl and stir thoroughly. Place the
bowl of water so that the water between the bottom of the bowl and
stir thoroughly. Bring to a boil, and leave some water in place of
the mixture so that the mixture is dry: If you do not like the texture
of the mixture, combine the milk before cooking.\n2) Remove the bowl
of water, and let it cool,
```

Figure 2: Recipe Generation by Baseline DistilGPT2

4 Baselines

4.1 Recipe Generation

The baseline model chosen for the task of Recipe Generation is DistilGPT2 (Sanh et al., 2019), a ≈ 82M parameter model, which was derived after performing Knowledge Distillation on GPT2. Knowledge Distillation is a process of reducing the size of a model, while maintaining similar accuracy to the original model. The training objective is to generate probability distributions as close to probability distribution of the original model. Similar to GPT2, DistilGPT2 is also a Transformer based model, which is pre-trained on the task of text generation. We chose DistilGPT2 due to the reduced amount of computational resources required for fine tuning the model (DistilGPT2 takes 1 hour for 1 epoch of fine tuning, whereas GPT2 takes 23 hours for 1 epoch). The baseline version is not finetuned on any recipe dataset, and is tested based on the knowledge it learned during pre-training (a sample generation is presented in Figure 2). The baseline model was tested using zero-shot prompting on the validation dataset (discussed in Section 4). The BLEU score and perplexity were then used to understand model performance. More details about the results and the hyperparameters are mentioned in Section 7.

4.2 Text Style Transfer

After the emergence of ChatGPT (Liu et al., 2023), it has been used as a baseline for a lot of research work due to its ability to generate coherent and contextually relevant responses. ChatGPT is a Large Language Model, which uses Transformer architecture with self-attention to generate unique and diverse outputs. It was pretrained on a large corpus of high quality data and was later fine tuned using Reinforcement Learning with Human Feedback (RLHF). The biggest advantage of this model is that it is easy to perform in-context learning (does not require updating the weights) by us-

```
INPUT: Re-write this recipe in the style of Shakespeare:
<Generated_Recipe>
```

Figure 3: Instruction Prompting for ChatGPT

ing various prompting techniques such as Few-Shot Learning (Parnami and Lee, 2022), Instruction Prompting (Prabhumoye et al., 2022), Chain-of-Though Prompting (Wei et al., 2023), and many more. Due to these advantages, it aligns well with the task of Text Style Transfer and acts as an upper bound baseline model. We utilize Instruction Prompting (Figure 3) as the in-context learning technique for generating styled text for a given input. For our example, we used an output from our finetuned recipe generation model. Results generated using the baseline and a quality comparison with our proposed model are discussed in Section 7.

5 Proposed Methodology

We are proposing a framework which performs the task of generating persona infused recipes, given some ingredients as an input. This framework consists of two main components - 1) Recipe generation and 2) Text Style Transfer (Figure 4). The main aim is to preserve the content generated in component 1 while infusing some personality into the text.

5.1 Recipe Generation

The first phase of the framework makes use of a pretrained Large Language Model (LLM), DistilGPT2, which is fine tuned on the preprocessed Recipe Generation dataset. There is a lot of research that questions whether training from scratch or fine-tuning is better. Usually, when we have a domain specific dataset that is large enough to make models learn useful representations, it is better to train the model from scratch. When dataset size or computational resources are limited, however, it is better to preform fine-tuning for a downstream tasks. Due to the limitation of computational resources, we have fine-tuned DistilGPT2 instead of training it from scratch. Recipe generation is a text generation task, so we make use of a decoder-only transformer model, which saves us a lot of computational power and memory compared to an Encoder-Decoder architecture. A list of ingredients $\mathbf{I} = [i_1, i_2, \ldots, i_n]$ is provided to the model as input, and a token of characters (generated by the model) joined together into a recipe paragraph p is the output.

5.2 Text Style Transfer

Recipes p generated by the transformer will be used as input to the next phase of our framework. Several pretrained Transformers will be fine-tuned in parallel using supervised learning on the preprocessed TST datasets to imitate and incorporate the style in the input text. We use p as an input to the finetuned Transformer models which, when passed through Encoders (using unmasked self-attention mechanism) $ENC_1, ENC_2, \ldots, ENC_n$, results in several contextualized latent space representations $z_1, z_2, \ldots, z_n$.

These intermediate latent space representations are then used by the Decoders $DEC_1, DEC_2, \ldots, DEC_n$. Each of them is fine tuned to output text with the same content c but styles $s_1, s_2, \ldots, s_n$, respectively (utilized cross-attention mechanism). More details about the personalities and datasets used are mentioned in Section 4.

During fine tuning, each encoder i will be fine tuned on X_i, where X_i is the corpus containing textual data that captures the style i of a celebrity. The encoder will learn to construct a contextualized latent space representation (final layer token level representation) of the input $x \in X_i$.

$$z = ENC_i(x, \theta_i) \tag{1}$$

This process results in a mapping between the corpus and their representation (X_i, Z_i). On the other hand, the decoder i for style i learns to reconstruct the original corpus X_i from the latent representation $z \in Z_i$ (learn how to generate a style infused text from the latent space).

$$\bar{x} = DEC_i(z, \theta'_i) \tag{2}$$

We are using T5-small as the Encoder-Decoder architecture for the Text Style Transfer. The two main reasons for choosing this model are as follows:

- Pretrained on the Paraphrasing/Sentence similarity datasets such as MRPC (Dolan and Brockett, 2005), STS-B (Cer et al., 2017), and QQP. This ensures that the model has some understanding of text style transfer.

- Comparatively fewer number of parameters ($\approx$ 60M) - fits into our computational budget.

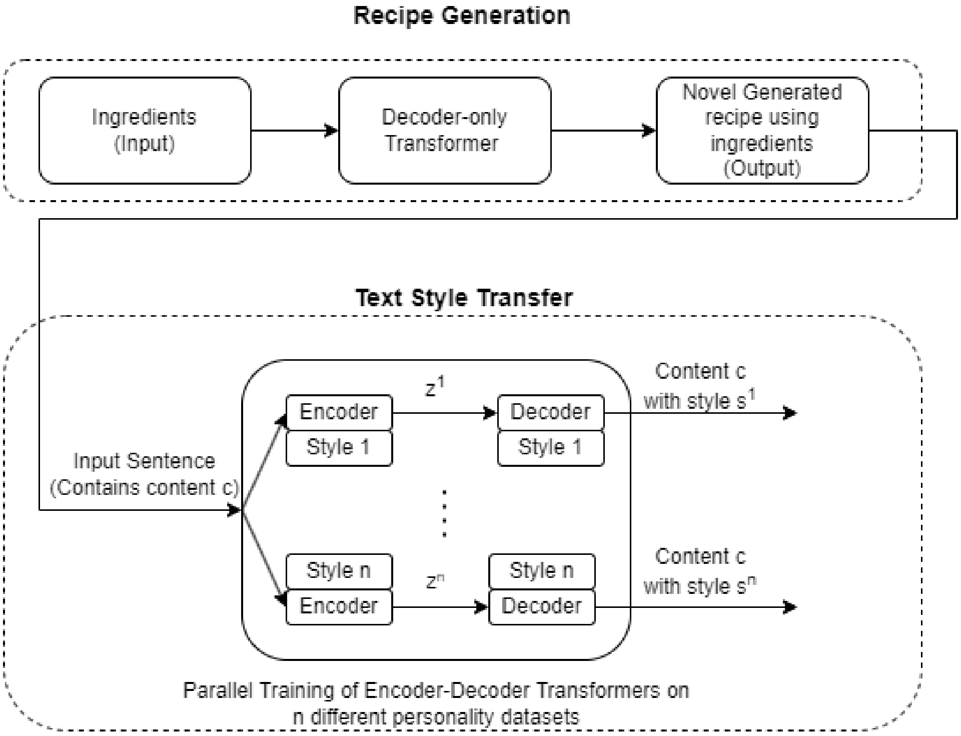

Recipe Generation
Ingredients (Input)
Decoder-only Transformer
Novel Generated recipe using ingredients (Output)
Text Style Transfer
Input Sentence (Contains content c)
Encoder
Style 1
z^1
Decoder
Style 1
Content c with style s^1
Style n
Encoder
z^n
Style n
Decoder
Content c with style s^n
Parallel Training of Encoder-Decoder Transformers on n different personality datasets

Figure 4: Recipe Infusion Framework

6 Experimentation and Results

6.1 Tools

In the dataset preprocessing phase of the project, we used Sklearn, Pandas, PyTorch and Numpy, which are libraries used for preprocessing/storing/loading of datasets. For the model development phase (and data tokenization), we made use of the transformers library from Hugging Face which allows us to use many pretrained models available for model development and baseline comparisons. We utilized DistilGPT2, T5-small, and finetuned MarianMT models for different phases of the project. Additionally, to measure the performance of the model on evaluation metrics we used evaluate library, which provides simple and easy to use implementations of the automatic evaluation metrics. Lastly, for visualizing the performance - training curves of the several models we used Matplotlib.

Code for all the tasks was implemented on Google Colaboratory. However, due to its computational restrictions, the model training was performed on a personal system which has following specifications:

- GPU: NVIDIA GeForce RTX 3080 Ti
- CPU: AMD Ryzen 9 5900X 12-Core Processor
- Memory: 32GB

6.2 Evaluation Metrics

Perplexity and BLEU scores are used to measure the performance of a text generation model. These metrics only judge the model based on content of the output, but they do not help explain the model's performance on style transfer. Therefore, an additional evaluation is required to measure the strength of style transfer. Hence, the three following fine-grained measures will be used independently:

- Semantic Preservation: For comparing/measuring similarity between the contents of input and output texts we will be using BLEU score (Papineni et al., 2002). There are many alternative metrics that are also used such as WordMover Distance (WMD) (Sato et al., 2022), sentence based Cosine similarity, and many more. These metrics capture the similarity between the reference and the generated output, and are good to judge semantic preservation.
- Fluency: This is one of the most important aspects of Natural Language model's outputs, and is measured using Perplexity. There

are many shortcomings noticed when judging a model based on perplexity: bias towards shorter sentences, different perplexity scores for words with the same meaning but different frequencies, etc. However, there is still no other automatic evaluation metric that is equally good/better than Perplexity in judging a model's fluency.

- Transfer Style Strength: To measure if a style transformer is able to transfer the style effectively, we perform Human Evaluation on the style transferred recipes. All the project members were asked to annotate $\approx$ 12 styled recipes for each persona, making it a total of 200 human annotated recipes (All the human annotators had gained enough knowledge about the different styles before annotations were done). More details about the Human annotation guidelines can be found in the Supplementary folder on Github.

6.3 Results

The proposed framework was successfully implemented in an end-to-end fashion. We have implemented all the code on our own, except for the models, which were accessed using Hugging Face.

6.3.1 Recipe Generation

In the first half, where the focus was given to Recipe Generation, DistilGPT2 was initialized using the Transformers library, and was fine tuned on the preprocessed recipe dataset. The model was trained for 20 epochs (Figure 5) using Adam optimizer (with default parameter values), with a learning rate of 5e-4 and weight decay constant of 0.01. Gradients were accumulated for 8 steps before performing any weight update step. Moreover, no updates were made for the first 1000 steps (called warm-up steps). The batch size was set to 8, which means that the model was trained on 8 different training set samples for each step. Lastly, we used a fixed precision training strategy where the model weights can have floating point values only up to 16 precise digits (this will definitely have an effect on the model's performance, but has a positive side of faster training due to less GPU space usage). The selection of hyperparameters such as batch size, fixed precision, and gradient accumulation during the training process was done considering our computational limitations.

The performance of our fine-tuned model and the baseline model on Recipe generation was measured using BLEU Score (Table 3) and Perplexity (Table 4). The Perplexity score was measured on the entire validation dataset ($\approx$ 65K samples), however, due to computational restriction we performed BLEU Score evaluation on only 5K samples. We see that the fine tuned model using a Greedy Decoding strategy performs better. Our fine-tuned model has a higher BLEU Score than the baseline model (irrespective of the decoding strategy used). An interesting insight is that the fine tuned model has a high 1-gram similarity with the reference recipes - much better than the baseline. This means that the model is generating coherent (semantically preserved) recipes. Surprisingly, the generations using Greedy Decoding are much better than the ones obtained by using Nucleus sampling. This may be because the BLEU score judges the similarity between the generated text and the reference, whereas the Nucleus sampling focuses on generating text with more diversity. The next evaluation metric, Perplexity, can be interpreted as the number of different words that can be used for the next prediction by the model. A lower perplexity is better, because it means that the model is fluent in the language generation task and is not considering unnecessary words. The fine-tuned model has a score of 4.1 (good), whereas the baseline model has a score of 28.58 (very bad). Therefore, we can conclude that the proposed model is better than the baseline model on the task of Recipe Generation (Quality difference between the generations by both the models can be observed by the examples provided in Section 8).

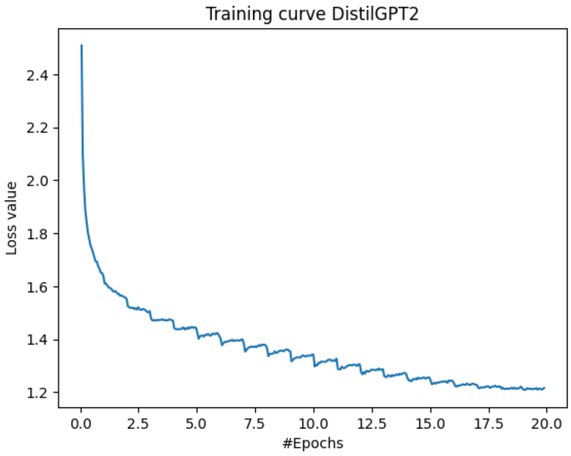


Figure 5: Training Loss Curve for DistilGPT2 (Recipe Generation)

	1-gram	2-gram	3-gram	4-gram	BLEU Score
DistilGPT2 (Greedy)	0.3696	0.3027	0.2822	0.2711	0.3041
DistilGPT2 (Ancestral)	0.3874	0.3050	0.2851	0.2746	0.3101
DistilGPT2 (Nucleus p=0.9)	0.3922	0.3121	**0.2923**	**0.2818**	0.3169
DistilGPT2 FT (Greedy)	**0.5212**	**0.3501**	0.2901	0.2595	**0.3424**
DistilGPT2 FT (Ancestral)	0.5122	0.3313	0.2742	0.2471	0.3275
DistilGPT2 FT (Nucleus p = 0.9)	0.5116	0.3287	0.2718	0.2450	0.3253

Table 3: BLEU Score performance of Recipe Generation models using different decoding strategies

6.3.2 Text Style Transfer

For Text Style transfer, we use the T5-small model (available through Transformers library), and fine tune it on the parallel datasets (Back Translated dataset and Identity translated dataset) using supervised learning. A different T5-small model was trained for each personality (in total 4 models were trained). Aside from the warmup steps and using a different batch size, all other hyperparameters used to fine tune DistilGPT2 were used to fine tune our T5-small model. We opted to use a batch size of 2, and did not use warm-up steps due to the smaller sample size.

We observed that when we trained the model on an Identity translated dataset, the output had no style during testing (failed/bad outcome). The model learned the identity mapping - replicate the input content (no additional style information was learnt). Therefore, we focused on using the back translated dataset. The Back Translation training loss curves for each persona are presented in Figure 6.

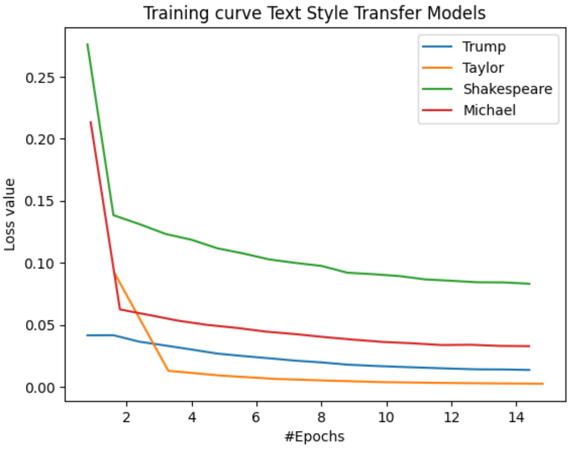


Figure 6: Training Loss Curve for different TST Models

As discussed in Section 5, we chose ChatGPT as the upper bound baseline model for TST. The proposed model was implemented with two different mechanisms - i) Entire recipe was given as an input to the model to infuse style and ii) each recipe was segmented into sentences, these sentences were styled one at a time and then concatenated to form a single, styled output recipe. When comparing the performance of the proposed model with ChatGPT for different personas, we found that ChatGPT was significantly better at style transfer. An example of style transfer by the models (input to the model was generated by fine tuned DistilGPT2 (Figure 7)) are presented in Figure 8-11, yellow highlighted text corresponds to stylistic attributes, whereas cyan highlighted text corresponds to addition of some non-meaningful information. As expected, ChatGPT was successfully able to infuse some style without losing any important content from the recipe. We used both Nucleus sampling and Greedy Decoding (results of only Greedy Decoding are presented), and the latter performed much better. Results portray that when sentences are styled individually, we observe better style transfer. Moreover, there is no addition of non-relevant text. Therefore, we perform Human evaluations on the styled recipes that are processed sentence-wise.

	Perplexity
DistilGPT2	28.58
DistilGPT2 FT	**4.10**

Table 4: Perplexity measure of DistilGPT2 and fine-tuned (FT) DistilGPT2

Human evaluation was done on 200 styled recipes (Table 5) which were generated by the fine-tuned DistilGPT2 model, and then styled by each fine-tuned T5-small model using greedy decoding (Recipes were sampled from the validation split). There is a significant performance gap when the model is trained on parallel data (62% - Shakespeare) versus when trained on non-parallel

	Lost Cause	Neutral/Tie	Successful
Donald Trump (NP)	8%	78%	14%
Michael Scott (NP)	6%	40%	54%
Taylor Swift (NP)	0%	70%	30%
Shakespeare (P)	6%	32%	62%

Table 5: Human Evaluation Results on the Styled Recipes

dataset (54% - Michael Scott) - which concludes that training on a truly parallel dataset will lead to better performance than training on a synthetic parallel dataset. A major part of TST is to preserve the content, and the model rarely (maximum - 8% for Donald Trump) adds style that also disrupts the input's content. Further, we now feel that style transfer on neutrally worded instructions such as recipes is a very hard task. It seems to be much easier to inject style into text that has some form of personality, but good instructions such as recipes aim to be very clear and unambiguous. Even in our best case benchmark using ChatGPT, we see that style was injected by adding additional sentences to the base recipe, rather than solely editing words within the input recipe.

7 Error analysis

An adversarial input is an input where the model deviates from its expected behavior due to some noise or out of distribution input. These inputs are used to check different failure cases of the model. In our project, we have four different failure cases for the DistilGPT2 (baseline) model:

1. Inclusion of non-food related objects/items as ingredients(ex. plastic). The model behaves completely illogically when such items are present in the input.

2. Very few (2) ingredients given as input (Non-zero) - In this case, the model usually tries to add new ingredients in the list, or it generates a recipe that does not make any sense at all.

3. Large number of ingredients given as input - The model's generated output makes sense for such situations, but usually it fails to use all the ingredients (or the recipe is incomplete, due to the restriction on maximum number of tokens to be generated). This usually occurs when the list is composed of > 10 ingredients.

4. Modification of the input pattern: More specifically, removal of the term 'Recipe:', from the input given to the model. Mostly, more ingredients are inserted, and a lot of repetitions are noticed along with no proper recipe given as an output.

When the fine-tuned DistilGPT2 model was tested on these inputs, we observed that the model was robust to all the adversarial cases - except when the input contained very few ingredient items, in this scenario the model adds new ingredients along with some irrelevant text such as links, and non-recipe related information. A comparison of the output generated by both the models is presented in Figures 12-15, where yellow highlighted part attributes to non-meaningful generation and cyan highlighted portion denoted the addition of extra ingredients. Our model performs superior to the baseline model, as it was able to generate a coherent and fluent recipe, which added extra ingredient/irrelevant information only rarely. A total of 120 such adversarial inputs were created which are available on the Github repository.

Error analysis for our text style transfer model has been covered in section 7.3.2 where human evaluation was conducted.

8 Contributions

- Quoc : Data collection/processing, Human Evaluation and Report

- Anshumaan : Built and trained models/pipelines, Human Evaluation, Report

- Ashvath : Data collection/processing, Model training, Human Evaluation, Report

- Saranath : Error analysis, Human Evaluation and Report

9 Conclusion

Some key takeaways from the project for our group has been getting hands on experience on topics such as Text Generation, Style Transfer, and Model Deployment. Model Development is a huge research area, but it is an equally important skill to learn how to deploy those models for customized downstream tasks. In our project, we needed to create a framework which utilizes two different models that have different configurations. This helped us understand different aspects of using these models with real world data(including areas such as their pretraining and evaluation). We also gained experience with hyperparameter tuning to fit the model and data while taking computational budget into account. This also involved different design choices for the framework, such as model selection, dataset size and data processing.

We also find that text style transfer is a more nuanced and difficult task than recipe generation. While the recipes themselves were not always edible or easy to follow, the natural flow of each of our recipes was impressive, and after training we had much better results than our baseline model. Our text style transfer model had a harder time producing interesting results, but as discussed, this may be due to a less than ideal input for style transfer. The lack of good parallel data was also a large bottleneck. We see that with properly paired data like the Shakespeare dataset, we were able to get better style transfer results than using our back translated datasets.

10 Future work

We believe that our first bottleneck comes from a relatively limited amount of compute power. Using bigger models and training for more epochs would certainly help us get better results in both recipe generation and text style transfer. Bigger models and better hardware could also allow for larger inputs, which may produce more natural sounding results. We were also required to train on a subset of our data due to speed and time limitations, so more compute power would help us train on much more data.

Our second bottleneck is due to a lack of good parallel data for style transfer. Because it is difficult to get parallel data for supervised training for this task, we believe a good next effort would heavily research better back translation implementations and parallel dataset synthesizers in general. While our backtranslation implementation did help our models learn some style, they were not as good as the model that learned on parallel data. One method we propose to generate high quality parallel datasets, is to use GPT4 or ChatGPT to produce styled outputs from un-styled inputs.

Finally, we can perform adversarial training to make our model robust to different input formats.

References

Batra, D., Diwan, N., Upadhyay, U., Kalra, J. S., Sharma, T., Sharma, A. K., Khanna, D., Marwah, J. S., Kalathil, S., Singh, N., Tuwani, R., and Bagler, G. (2020). RecipeDB: a resource for exploring recipes. *Database*, 2020. baaa077.

Bień, M., Gilski, M., Maciejewska, M., Taisner, W., Wisniewski, D., and Lawrynowicz, A. (2020). RecipeNLG: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28, Dublin, Ireland. Association for Computational Linguistics.

Carlon, K., Riddell, A., and Rockmore, D. (2018). Evaluating porse style transfer with the bible. *arXiv:1711.04731 [cs.CL].*

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. (2017). SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., and Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Dolan, W. B. and Brockett, C. (2005). Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005).*

Fu, Z., Tan, X., and Zhao, D. (2017). Style transfer in text exploration and evaluation. *arXiv:1711.06861 [cs.CL].*

Goel, M., Chakraborty, P., and Ponnaganti, V. (2022). Ratatouille: A tool for novel recipe generation. *arXiv:2206.08267 [cs.CL].*

Hu, Z., Lee, R., and Aggarwal, C. (2022). Text style transfer: a review and experimental evaluation. *arXiv:2010.12742 [cs.CL].*

Iyyer, M., Wieting, J., and Krishna, K. (2020). Reformulating unsupervised style transfer as paraphrase generation. pages 737–762.

Jhamtani, H., Gangal, V., and Hovy, E. (2017). Shakespearizing modern language using copy-enriched sequence-to-sequence models. *arXiv:1707.01161 [cs.CL].*

Jin, D., Jin, Z., and Hu, Z. (2022). Deep learning for text style transfer: A survey. *Computational Linguistics, Volume 48, Issue 1 - March 2022.*

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., and Socher, R. (2019). Ctrl: A conditional transformer language model for controllable generation.

Lin, K., Liu, M., and Sun, M. (2020). Learning to genrate multiple style transfer outputs for an input sentence. *arXiv:2002.06525 [cs.CL].*

Liu, Y., Han, T., Ma, S., Zhang, J., Yang, Y., Tian, J., He, H., Li, A., He, M., Liu, Z., Wu, Z., Zhu, D., Li, X., Qiang, N., Shen, D., Liu, T., and Ge, B. (2023). Summary of chatgpt/gpt-4 research and perspective towards the future of large language models.

Marin, J., Biswas, A., Ofli, F., Hynes, N., Salvador, A., Aytar, Y., Weber, I., and Torralba, A. (2019). Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Parnami, A. and Lee, M. (2022). Learning from few examples: A summary of approaches to few-shot learning.

Prabhumoye, S., Kocielnik, R., Shoeybi, M., Anandkumar, A., and Catanzaro, B. (2022). Few-shot instruction prompts for pretrained language models to detect social biases.

Prabhumoye, S., Tsvetkov, Y., and Salakhudinov, R. (2018). Style transfer through back-translation. *arXiv:1804.09000.*

Pryzant, R., Martinez, R., and Dass, N. (2018). Automatically neutralizing subjective bias in text. *arXiv:1711.04731 [cs.CL].*

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Salvador, A., Drozdzal, M., and Romero, A. (2019). Inverse cooking: Recipe generation from food images. *arXiv:1812.06164 [cs.CV].*

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS* $EMC^2 Workshop$.

Sato, R., Yamada, M., and Kashima, H. (2022). Re-evaluating word mover's distance.

Shen, T., Barzilay, R., and Jaakkola, T. (2017). Style transfer from non-parallel text by cross-alignment. *arXiv:1705.09655 [cs.CL].*

Syed, B., Verma, G., and Natarajan, A. (2020). Adapting language models for non-parallel author-stylized rewriting. *arXiv:1909.09962 [cs.CL].*

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Huggingface's transformers: State-of-the-art natural language processing.

11 Appendix

Ingredients: [rice, egg, chicken, shrimp, soy sauce, sesame oil, salt, pepper, green onion] Recipe:

Cook rice and set aside. In a large nonstick frying pan, add olive oil and heat until almost smoking. Add chicken/seitan and cook covered for 10 minutes. Add shrimp and cook for 2-3 minutes. Add egg, cook until egg is cooked (scrambled). Add rice and stir up; add soy sauce and seasonings. Garnish with chopped scallions and/or crushed red pepper. Serve hot. Enjoy. Makes 4-6 servings as a main dish. Cooking time is 15-20 minutes. Enjoy. Serves 2.5.

Figure 7: Recipe Generated by FT DistilGPT2 model (will be used for comparison of baseline with fine tune TST model)

Alright, people, listen up! Today, we're gonna make some rice dish that's gonna blow your socks off! Cook that rice and put it aside, we don't need it just yet. Now, take a big nonstick pan and pour some olive oil in there. Heat it up until it's hot enough to set off the smoke alarm! Now, let's add some chicken or seitan, cover it up and let it cook for 10 whole minutes. Next, throw in some shrimp and cook for just a couple of minutes - we don't want them to get overcooked, do we? Now, crack an egg in there, scramble it up real good, and cook it until it's nice and firm. Alright, it's time to bring in that rice we set aside earlier. Mix it all together, pour some soy sauce in there, and add some seasoning to taste. And finally, let's top it off with some chopped scallions and crushed red pepper to really give it that zing! And that's it, folks! Serve it up hot and enjoy the heck out of it! This recipe makes enough for 4-6 people, but who am I kidding, you'll probably want it all to yourself! And it only takes 15-20 minutes to cook. So what are you waiting for? Get cookin'!

(a) Styled Recipe Generated by ChatGPT

cook rice and set aside. in a large nonstick frying pan, add olive oil and heat until almost smoking. add some chicken/seitan and cook covered for 10 minutes. add shrimp and cook for 2-3 minutes. Add egg, cook until egg is cooked (scrambled). Add rice and stir up; add soy sauce and seasonings. Garnish with chopped scallions and/or crushed red pepper. Serve hot. Enjoy. makes 4-6 servings as a main dish. Cooking time is 15-20 minutes. Enjoy. Serve hot. Enjoy. Serve hot. Enjoy. Serves 2.5.

(b) Style Infused on Entire Recipe

cook rice and set aside. in a large pan nonstick, add olive oil and heat until almost smoking. add some chicken/seitan and cook covered for 10 minutes. add shrimp and cook for 2-3 minutes add egg. cook until egg is cooked (scrambled) add rice and stir up; add soy sauce and seasonings. gimme with chopped scallions and/or crushed red pepper. Serve hot oh, ok. ok. ok. ok. ok. ok. ok. makes 4-6 servings as a main dish. cook time is 15-20 minutes oh, ok. ok. ok. ok. ok. ok. ok. oh, god. Serves 2 people. 5 oh, ok.

(c) Sentence-wise Styling

Figure 8: Comparative Evaluation - *Michael Scott*: ChatGPT vs. Proposed Model w/ Full Input vs. Proposed model w/ sentence level input.

Step into the kitchen, it's time to cook
Gonna make a dish that'll make you shook
First, cook up some rice, set it aside
In a nonstick pan, pour in olive oil with pride
Heat it up 'til it's almost smoking hot
Then add the chicken or seitan, let it cook with the pot
Cover it up for ten whole minutes
Then throw in the shrimp, don't leave it timid
Let it cook for two to three
Next up, crack in an egg, it's the key
Scramble it up until it's cooked through
Now it's time to add the rice, don't be slow to do
Stir it all up, make sure it's mixed well
Pour in soy sauce and seasonings, can't you tell?
Garnish with some scallions or red pepper crush
This meal is gonna make you blush
Serves four to six, it's a main dish
Cooking time's only 15-20 minutes, that's delish
Enjoy every bite, savor the flavor
Taylor Swift's recipe, it'll do you a favor

(a) Styled Recipe Generated by ChatGPT

cook rice and set aside in a nonstick frying pan, add olive oil and heat till almost smoking. Add chicken and sea salt and cook for 2-3 minutes (scrambled). Garnish with chopped scallions and/or crushed red pepper. Serve hot. Serve hot. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serves 2.5. Serve.)) scallions and samarite

(b) Style Infused on Entire Recipe

tossing rice and set aside in a large nonstick frying pan, add olive oil and heat till almost smoking add chicken/seitan and cook covered for 10 minutes add shrimp and cook for 2-3 minutes add the egg, cook till it's cooked (scrambled) add rice and stir up; add soy sauce and seasonings Garnish with chopped scallions and/or crushed red pepper Serve hot genie-fin' makes 4-6 servings as a main dish cooking time is 15-20 minutes genie-fin' serves 2 people 5

(c) Sentence-wise Styling

Figure 9: Comparative Evaluation - *Taylor Swift*: ChatGPT vs. Proposed Model w/ Full Input vs. Proposed model w/ sentence level input.

Listen folks, we're going to make the best rice dish ever, believe me! First, you gotta cook the rice, set it aside, and make sure it's the best rice you've ever seen. Then, in a huge, and I mean HUGE nonstick frying pan, add some olive oil, and heat it up until it's almost smoking hot, just like my rallies!

Now, you're gonna add some chicken, or if you're into that vegan thing, seitan, and you're gonna cook it covered for 10 whole minutes, it's gonna be great, trust me! Then, you add some shrimp, and let it cook for 2-3 minutes, it's gonna be fantastic!

Next, you crack open an egg, scramble it up real good, and cook it until it's cooked, just like how we're gonna cook China in the trade war! Then, you add the rice, and mix it up really well, we want the rice to be coated with all the flavors, it's gonna be tremendous!

Finally, you're gonna add some soy sauce and seasonings, it's gonna be so good, so good, you're not even gonna believe it! We're gonna garnish it with some chopped scallions and some crushed red pepper, it's gonna be a beautiful dish, really beautiful! Serve it hot, and enjoy! This dish makes 4-6 servings as a main dish, it's gonna be huge, folks! And cooking time is just 15-20 minutes, it's gonna be quick, easy, and delicious, believe me! And it serves 2.5, it's not gonna be a loser, folks!

(a) Styled Recipe Generated by ChatGPT

heat well and set aside. in a large nonstick pan, add olive oil and heat until almost smoke. add chicken/seitan and cook for 2-3 minutes. add shrimp and cook for 2-3 minutes. add soy sauce and seasonings. Serve hot. enjoy. makes 4-6 servings as a main dish.

(b) Style Infused on Entire Recipe

cook rice and set aside in a great nonstick pan, add olive oil and heat until almost a year add chicken/seitan and cook covered for 10 minutes add shrimp and cook for 2-3 minutes add egg, cook until egg is cooked (scrambled) add rice and stir in; add soy sauce and seasonings trump torn up with chopped scallions and/ou crushed red pepper serve hot enjoy “makes 4-6 servings as a main dish” cooktime is 15-20 minutes enjoy servitude 2 5

(c) Sentence-wise Styling

Figure 10: Comparative Evaluation - *Donald Trump*: ChatGPT vs. Proposed Model w/ Full Input vs. Proposed model w/ sentence level input

Hark! First, cooketh the rice and set it by,

In a grand nonstick frying pan, olive oil addeth, and heat until almost smoking high.

The chicken or seitan in, cover and cook for ten,

The shrimp toss in, cook for two or three, and then

The egg cracked in, scramble it well,

Add the rice and stir until it all jell.

Soy sauce and seasonings, give it a good toss,

Garnish with scallions and crushed red pepper to enhance the sauce.

Serve it hot and relish the flavors divine,

As a main dish for four to six, its portion's quite fine.

Enjoy thy meal, forsooth, for it doth take only 15-20 minutes to dine,

And knoweth, this recipe doth serve 2.5, verily it is divine.

(a) Styled Recipe Generated by ChatGPT

cook rize and set aside. in a large nonstick pan, add olive oil and heat till almost savor. add chicken and cook for 10 minutes. add shrimp and cook for 2-3 minutes. add rice and stir; add soy sauce and seasonings; scallions and garlic to sauce; serve hot. oh, make 4-6 servings for the main dish: cook 15-20 minutes. eat.

(b) Style Infused on Entire Recipe

cook the rize and set aside. add olive oil and heat in a large pan, till almost taint the fumitive smoke. add chicken/seitan and cook covered 10 minutes. add shrimp and cook 2-3 minutes. add egg, cook till egg be cooked (scrambled) add rice, and stir up; add soy sauce and seasonings; scallions and/or crushed roes of garlic, snatch and scallions, serve hot. o, joy, and joy. makes 4-6 servings for the main dish. cook 15-20 minutes. o, joy, and joy. serves 2 five.

(c) Sentence-wise Styling

Figure 11: Comparative Evaluation - *William Shakespeare*: ChatGPT vs. Proposed Model w/ Full Input vs. Proposed model w/ sentence level input.

1 cup of vanilla extract\n1 tablespoon of ciander / vanilla extract Instructions Preheat the oven to 300-200 F. In a 350F oven, bring the milk, milk, and vanilla extract into a bowl. Slowly whisk together the spinach, mozzarella, oregano, and bay leaves or chopped ginger. Add the green juice and the remaining coconut milk, then stir to combine the spinach with the milk and juice. Remove from the oven and let cool for 40-45 minutes.\nPlace a tray on a baking sheet and place it in an oven until golden brown and completely golden brown. The topping will look nice on the surface

(a)

Whiz everything together in a blender until well blended and it becomes a thick spread. Make sure you mix all the ingredients in a blender as this makes a smooth spread. Spread that mixture on to the bread. Place under the broiler about 4 minutes, or until the cheese browns and bubbles. Serve. Note: you may be able to slice these bread into about 10 pieces and spread them out a bit so they won't dry out. Also, the spread makes a nice finishing touch if you don't go big around with a spoon. Enjoy. Yum! :)!!!!! http://www.therealb

(b)

Figure 12: Adversarial Inputs - *input format is changed*: baseline (a) vs. FT model (b)

[fresh green apples] Recipe: [chocolate] Recipe: [sauce] Recipe: Recipe: [sauce] Recipe: [sauce] Recipe: [sauce] Recipe: [sauce] Recipe: [sauce] Nutrition Facts Nutritional Fat: 10g / grams Calories from Fat - Total Fat 1g / grams Fiber - Fat 6g / grams Fat - Total Fat 1g / grams Fiber - Fat 9g / grams Fiber - Fat 6g / grams Fat - Total Fat 1g / grams Fiber - Fat 5g / grams Fiber - Total Fat 1g / grams Fiber - Fat 6g / g Fiber

(a)

Combine ingredients in a bowl and let stand at room temperature for 24 hours. Notes: Substitute orange juice for white grape juice with orange juice. 1 1/2 jasmine tea bag, loosely packed. Pour a tsp or so into each cup of a teacup and fill with tequila. Serve chilled with or without a garnish of fresh fruit and sugar. Yield: 2 1/2 cups or 32 servings (approx. 0 cup per serving).   Read More http://www.ethanks-basting.com/recipe/recipe/ugget-cups.html?oc=linkback.AVier-v favoring

(b)

Figure 13: Adversarial Inputs - *number of ingredients in are too less*: baseline (a) vs. FT model (b)

[2] Prep Time 10 minutes Servings - 3 2 - 4 Calories from Fat - 30 kcal Author : Marcel Gersberg\n\n\n\n\n\n\n\n\n\n- Foul of the recipe\n\n\n\n\n\nHow many times do I have to use

(a)

Bring the vinegar to the boil in a small saucepan, remove from the heat and cool to room temperature. Cook the pasta in a large pan of salted boiling water according to the packet directions. Add 3 tbsp (45ml) of the cooked pasta into the pan, and cook for

(b)

Figure 14: Adversarial Inputs - *Excess number of ingredients*: baseline (a) vs. FT model (b)

[recipe, sour cream, pumpkin juice, ice cream, pumpkin jam, lime (not added), pumpkin pie\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\nTo order this recipe, you can either use a recipe from this cookbook or create a separate page to add the recipe to the page. For example, my recipe for dressing ice cream, frozen pumpkin pie and pineapple pie is the link for this recipe.\nThis recipe can either be used by your own kitchen for your own baking and pie recipes, or by a kitchen for your own baked goods for your own kitchen, or by someone

(a)

Combine all ingredients in a large punch bowl and punch bowl. If you have trouble getting punch right down and you need something to help you have to do with. If you don't have Aguher, you might want to add sugar or sugar. Pour punch over crushed ice into a punch bowl and enjoy punch. I add an extra splash of lime juice to the punch bowl. Serves 50. This can easily be halved and put in a punch bowl. Can be made in large quantities and frzn. Toss lightly but do not make too thick. Serves 60. Can be doubled. Enjoy. Serves 100. Food exchange.

(b)

Figure 15: Adversarial Inputs - *non-edible items are present in ingredient list*: baseline (a) vs. FT model (b)