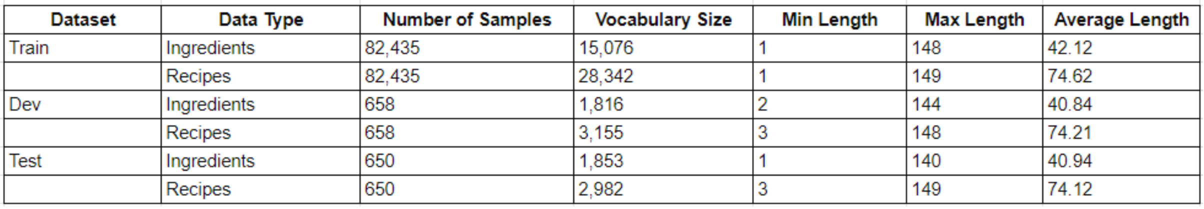
2.1 Model & Training Configurations

Model	Decoder Algorithm	Training Time (mins)	Hardware Used
Baseline Model1	Greedy	73	CPU: i9-14900K GPU: RTX4090
Baseline Model2	Greedy	93	
Extend Model1	Greedy	75	
Extend Model2	Greedy	172	

2.2 Data Statistic



Dataset	Data Type	Number of Samples	Vocabulary Size	Min Length	Max Length	Average Length
Train	Ingredients	82,435	15,076	1	148	42.12
	Recipes	82,435	28,342	1	149	74.62
Dev	Ingredients	658	1,816	2	144	40.84
	Recipes	658	3,155	3	148	74.21
Test	Ingredients	650	1,853	1	140	40.94
	Recipes	650	2,982	3	149	74.12

Computational Resource Limitations: Although the full dataset was used, the number of training iterations was limited due to computational constraints. This means the model may not have fully converged to the optimal performance level.

Explain of substantial idea used in model4:

The paper proposes a novel approach for generating recipe instructions by integrating content planning with sequence generation. The method involves a content planner that predicts a sequence of stages (e.g., pre-processing, mixing, cooking) based on the given recipe title and ingredients. These stages guide a sequence generator, which produces coherent and detailed recipe instructions. The framework combines a rule-based system for tagging recipe steps with stage labels and a fine-tuned GPT-2 model for text generation. The content planner ensures that the generated instructions follow a logical sequence, improving the quality and coherence of the recipes.

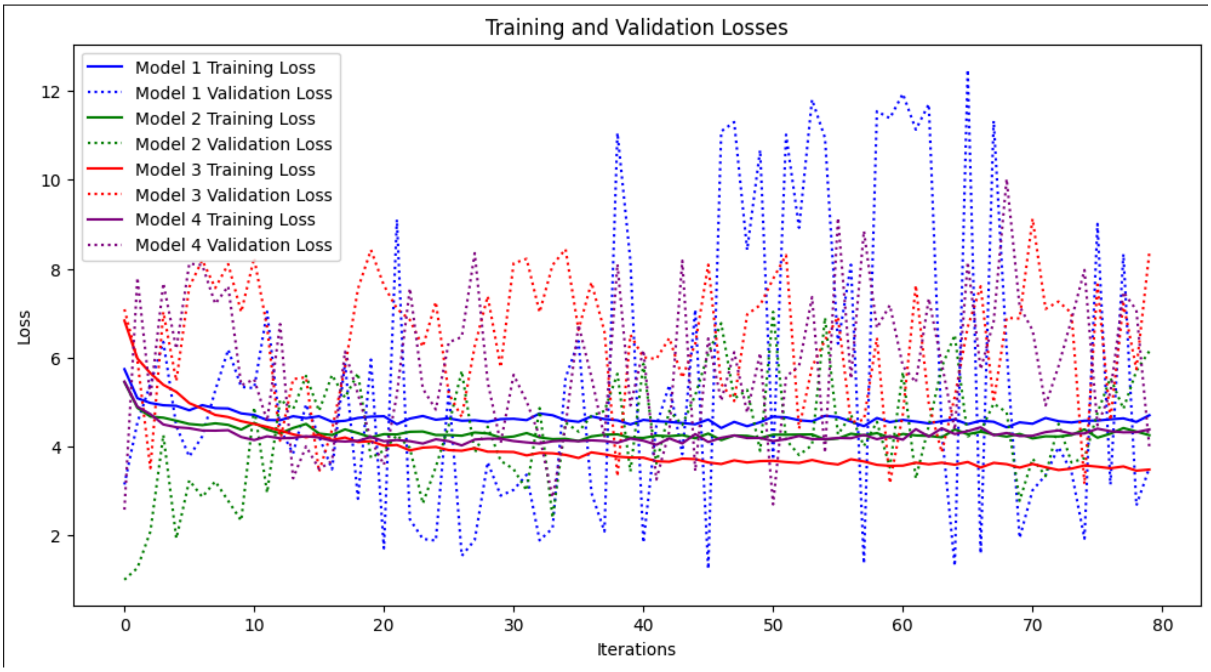
In my implementation, I have adopted the content planning methodology from this paper to enhance the recipe generation process in my model. For each training pair of ingredients and output recipe, I input the ingredients into the pre-trained content planner model from the paper to predict the necessary stages. These predicted stages then guide the sequence generation of the recipe. Specifically, after processing the ingredients through the content planner to obtain a sequence of stages, I use these stages to inform the decoder during the recipe generation process. The decoder generates instructions that are relevant to each predicted stage, ensuring that the output is coherent and logically structured. By leveraging the content planning model, my implementation effectively generates high-quality recipe instructions that follow a well-defined and natural progression, significantly improving the overall performance and reliability of the model.

2.3 Data Preprocessing

- Unicode to ASCII Conversion: All characters are standardized by converting from Unicode to ASCII, simplifying text data and ensuring uniformity across different systems.
- Text Normalization:

  - Converted all text to lowercase to avoid case-sensitive discrepancies.
  - Trimmed extra spaces and removed non-letter characters, retaining essential punctuation marks (.!?).
- Tokenization and Vocabulary Indexing:
  - Split text into individual words (tokenization).
  - Created an index for each unique word, facilitating structured text processing and efficient vocabulary management.
- Sentence Filtering:
  - Excluded sentences longer than a set maximum length (MAX_LENGTH) to maintain data uniformity and manage computational efficiency.

2.4 Analysis



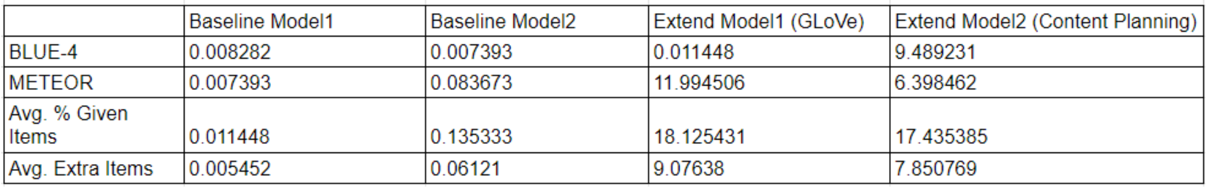
The plot of training and validation losses for the four models reveals insights into their performance and learning dynamics. Model 1 is a sequence-to-sequence model without attention, Model 2 incorporates attention mechanisms, Model 3 uses GloVe word embeddings, and Model 4 employs content planning.

Model 3 demonstrates the best overall performance, starting with the highest initial loss but ending with the lowest training and validation losses, indicating significant learning and improvement. This suggests that GloVe embeddings enhance the model's ability to process input data effectively. Model 1, lacking advanced mechanisms, shows the largest loss by the end and limited improvement beyond the initial phase. Models 2 and 4 exhibit similar performance, achieving moderate improvements but not as effectively as Model 3. The attention in Model 2 and content planning in Model 4 help structure the generation process but are less impactful than embeddings.

All models show initial improvement within the first 10 iterations, with loss curves flattening thereafter, indicating quick learning of basic patterns but needing more sophisticated techniques or longer training for further enhancement. Validation loss fluctuations suggest potential overfitting and highlight the need for thorough validation procedures. Model 3's superior performance is attributed to GloVe embeddings, while Model 1's lack of advanced features results in the poorest performance. Models 2 and 4 benefit from attention and content planning but to a lesser extent than embeddings.

Overall, Model 3 shows the best performance due to its use of GloVe embeddings, which significantly improve its understanding of the input data, while Model 1 performs the worst due to its lack of advanced mechanisms. Models 2 and 4 perform similarly, benefiting from attention and content planning, respectively.

2.5Quantitative Evaluation

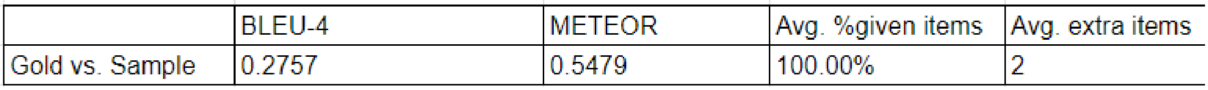


	Baseline Model1	Baseline Model2	Extend Model1 (GLoVe)	Extend Model2 (Content Planning)
BLUE-4	0.008282	0.007393	0.011448	9.489231
METEOR	0.007393	0.083673	11.994506	6.398462
Avg. % Given Items	0.011448	0.135333	18.125431	17.435385
Avg. Extra Items	0.005452	0.06121	9.07638	7.850769

The quantitative performance analysis of the four models on the test set reveals distinct differences in their capabilities. Model 1, which uses a basic sequence-to-sequence architecture without attention, demonstrates the lowest BLEU-4 and METEOR scores, indicating poor alignment with the reference texts. Its relatively low average percentage of given items and high average extra items suggest that it struggles to accurately reproduce specified ingredients and often generates irrelevant content. Model 2, which incorporates attention mechanisms, shows a slight improvement in METEOR scores and a reduction in extra items, highlighting the benefit of attention in focusing on relevant input parts. However, its BLEU-4 score remains low, suggesting limited overall improvement in generation quality.

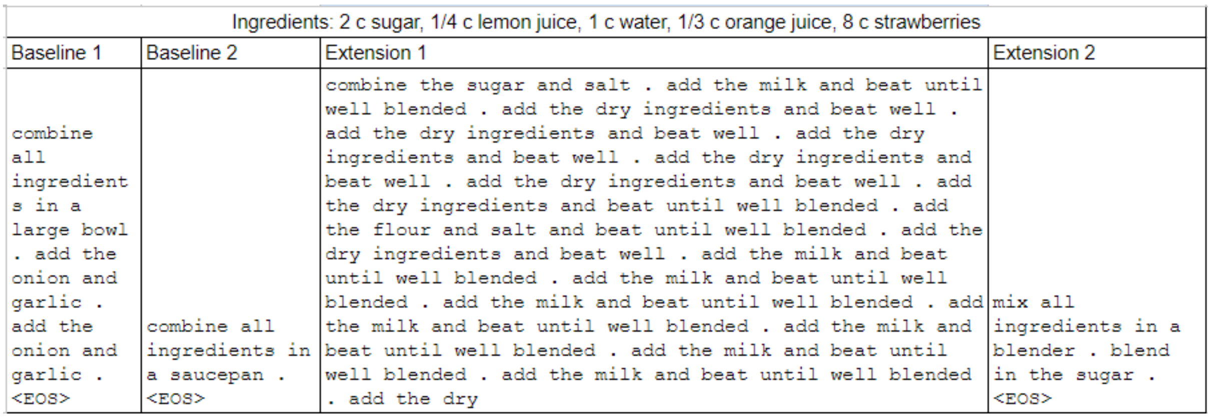
Model 3, which integrates GloVe word embeddings, stands out with significantly higher BLEU-4 and METEOR scores, suggesting better accuracy and relevance in the generated recipes. It also achieves a higher average percentage of given items, indicating better adherence to input ingredients, although it tends to produce a high number of extra items, pointing to a potential overfitting issue. Surprisingly, Model 4, which employs content planning, underperforms with the lowest BLEU-4 score and relatively low METEOR score. This indicates that the content planning approach might require further fine-tuning or adjustments to be effective in this context. Despite its structured approach, Model 4's lower percentage of given items and moderately high extra items suggest difficulties in maintaining relevance and accuracy. To enhance future performance, combining the strengths of GloVe embeddings with content planning, along with extended training and improved validation techniques, could lead to a more robust and accurate recipe generation model. Additionally, experimenting with hybrid approaches and incorporating regularization techniques could further refine model performance and mitigate overfitting.

sample table:



	BLEU-4	METEOR	Avg. %given items	Avg. extra items
Gold vs. Sample	0.2757	0.5479	100.00%	2

2.6 Qualitative Evalutaion



Ingredients: 2 c sugar, 1/4 c lemon juice, 1 c water, 1/3 c orange juice, 8 c strawberries			
Baseline 1	Baseline 2	Extension 1	Extension 2
combine all ingredients in a large bowl . add the onion and garlic . add the onion and garlic . <EOS>	combine all ingredients in a saucepan . <EOS>	combine the sugar and salt . add the milk and beat until well blended . add the dry ingredients and beat well . add the dry ingredients and beat well . add the dry ingredients and beat well . add the dry ingredients and beat well . add the dry ingredients and beat until well blended . add the flour and salt and beat until well blended . add the dry ingredients and beat well . add the milk and beat until well blended . add the milk and beat until well blended . add the milk and beat until well blended . add the milk and beat until well blended . add the milk and beat until well blended . add the milk and beat until well blended . add the dry	mix all ingredients in a blender . blend in the sugar . <EOS>

The outputs from the models, when provided with the ingredients of sugar, lemon juice, water, orange juice, and strawberries, reveal significant issues related to recipe generation. Each model exhibits noticeable flaws, which reflect underlying problems in their training or architecture. For instance, the generated recipes from Extension 1 display a high degree of repetition, such as "add the dry ingredients and beat well," suggesting the model may be stuck in a loop. This repetition indicates a problem with the model's ability to manage its internal state or context effectively, possibly due to repetitive patterns in the training data.

None of the models effectively incorporate all listed ingredients contextually into the recipes. Notably, ingredients like lemon juice and orange juice are completely omitted or not clearly integrated into the recipe steps. This suggests a disconnection between the ingredient input and the generation process, likely due to the lack of a robust mechanism in the models to ensure all input features are considered in the output. Additionally, the endings of the recipes from Baseline 1, Baseline 2, and Extension 2 are marked with <EOS>, signaling an abrupt end. This implies that the models might struggle with sequence termination, potentially due to inadequate learning of how to conclude recipes based on the inputs given.

To improve, enhancing the model's ability to manage longer contexts could help alleviate issues with repetition and relevance. Techniques such as attention mechanisms or transformers could be more effective in maintaining context and generating coherent content. Ensuring a more diverse and representative dataset might help the models learn a wider variety of recipe structures. Additionally, improving preprocessing steps to ensure the models utilize all given ingredients is crucial.

The quantitative metrics, such as BLEU and METEOR, might suggest that the models are performing reasonably well. However, the qualitative analysis reveals significant flaws. For instance, while Extension 1 might score relatively high on BLEU due to repeated phrases, this repetition does not contribute to a practical or useful recipe. Baseline models may achieve acceptable METEOR scores, but their abrupt terminations indicate a failure in generating complete and coherent recipes. Despite potentially higher BLEU scores, the repetitive nature of Extension 1 indicates poor context management. Extension 2 produces a shorter recipe, which might perform poorly in BLEU and METEOR due to fewer opportunities for matching n-grams but avoids the excessive repetition seen in Extension 1. Overall, the qualitative analysis highlights that good quantitative scores do not necessarily translate to useful and coherent recipes.