FIT5217 Assignment 2
Student Name: YI JIE NG
Student ID: 31158145

Model Specification:

Baseline 1: Sequence-to-sequence RNN
Baseline 2: Sequence-to-sequence RNN with attention
Extension 1: Sequence-to-sequence RNN with attention and data-preprocessing
Extension 2: Sequence-to-sequence RNN with attention, data-preprocessing, pretrained word2vec embedding and Paper: Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, Yejin Choi. NeuroLogic Decoding: (Un)supervised Neural Text Generation with Predicate Logic Constraints. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics.

Detailed Descriptions from the paper chosen:

The core technique mentioned in the paper emphasizes on predicate logic constraints where D(a,y) will be true if a occurs in y. The final formula obtained is shown as:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} P_\theta(\mathbf{y}|\mathbf{x}) - \lambda' \sum_{i=1}^{L}(1 - C_i)$$

In our ingredient-recipe generation task, this idea can be translated to an objective function: argmax(ingredients in recipe predicted = ingredients given AND no extra ingredients).

To apply the high-level idea of the technique mentioned to our model, penalty and compensation loss have been introduced to our model when calculating the loss. To match the first constraint in objective function, we subtract a compensation value (current loss*0.5) to the original loss to **slightly decrease the total loss**. Same goes to the second constraint from the objective function, we add a high penalty value (current loss*1.5) to the original loss to **increase the total loss**.

By doing this, model is expected to generate the ingredient given in the recipe as much as possible at the same time reduce the extra ingredients.

2.1 Model & Training Configurations

	Baseline 1	**Baseline 2**	**Extension 1**	**Extension 2**
RNN cell	LSTM			
Hidden size	256			
Teacher forcing ratio	1.0			
Optimizer	Adam			
Dropout rate	0.1			
Maximum length	150			
Word embeddings dimensionality	300			
Training iteration	10000			
Learning rate	0.001			
Training time	34mins 47s	44mins 47s	36mins 48s	21mins 48s
Decoding algorithm	Greedy algorithm by choosing the word with highest probability			
Hardware detail	RTX 3070 with 8gb VRAM using cuda			

FIT5217 Assignment 2
Student Name: YI JIE NG
Student ID: 31158145

2.2 Data Statistics

	Number of samples	Vocabulary size	Min length	Max length
Ingredients without preprocessing	79434 train pairs, 641 dev pairs, 620 test pairs	107361	1	149
Recipe without preprocessing		35037	1	149
Ingredients with preprocessing	87770 train pairs, 695 dev pairs, 685 test pairs	14479	1	144
Recipe with preprocessing		27381	1	149

- *Training data used portion: 100%. All of the training data have been.*
- *Ingredients and recipe all indicate to the same train, valid and test set*

2.3 Data Preprocessing

For every model, I have removed any data having no value (either ingredients or recipe or both), then convert the cell value into string. For extension 1 and 2 model with data preprocessing:

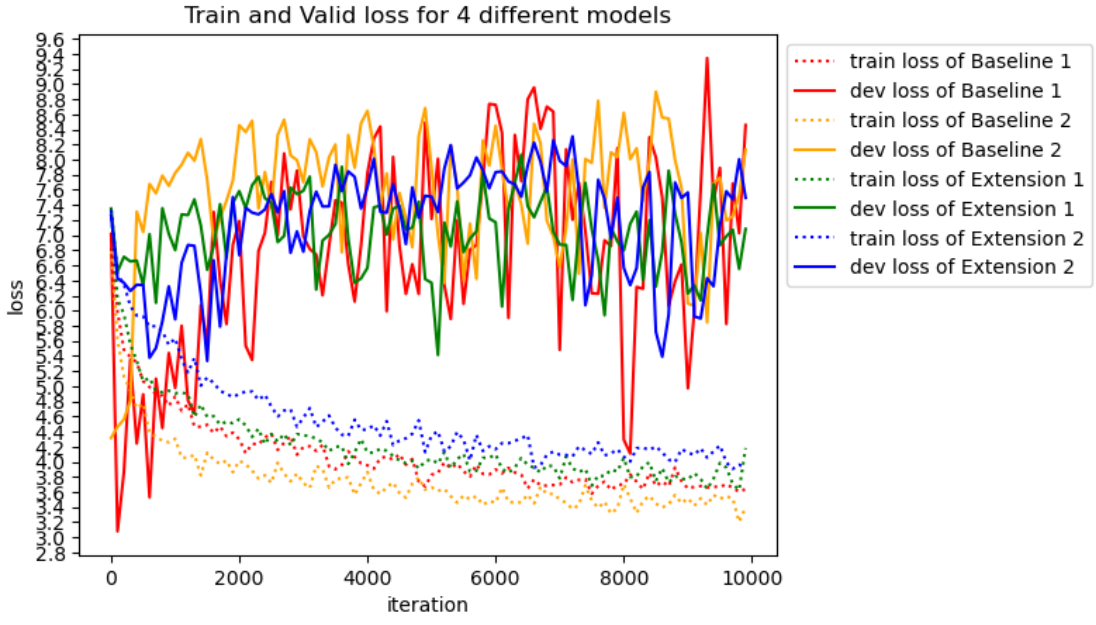
For ingredients: 1) Lower down every character and transform them from Unicode to ascii. 2) Substitute the stopwords with space. 3) Remove character other than a to z and space notation such as \t. 4) Remove common non-ingredients words. 5) Separate the left valid ingredients (using \t). 6) remove the extra space. 7) Make the repeated ingredients to a unique ingredient to avoid repeated ingredients. 8) Concatenate the unique ingredients into a long string.

For recipe: First three steps of ingredients.

The reason why ingredients have extra preprocessing step compared to recipe is because recipe is considered as ground truth label, and the expected output of model should be a valid and complete sentence of a recipe. However, we only need a certain information for ingredient thus more preprocessing step is introduced for ingredients compared to recipe.

2.4 Analysis

a. Train and Valid loss for each model



b. Test loss for each model

	Baseline 1	**Baseline 2**	**Extension 1**	**Extension 2**
Test loss	9.0422	6.3945	7.3503	7.9582

c. Justification

Generally, we use **train loss** to fit and let the model **learn** to minimize the loss. The training loss for all models **keep decreasing and slowly converged**. The purpose of **valid loss is to evaluate the performance** of the models. We then **modify the hyperparameter** and manually trying to minimize the valid loss. Finally, **test loss** is used for evaluating the **performance of trained model** with defined hyperparameter. Based on the plotting and the test loss evaluated, we can conclude that **Baseline 2 model outperformed all of the models, followed by Extension 1 and 2** model. The **worst model** that computes the highest test loss goes to **Baseline 1**.

As we can see from the graph for best model of **Baseline 2, the valid loss** has the **most obvious decreasing trend**. This is because since baseline 2 has no text preprocessing but with attention mechanism, there are still useful information or relationship available for model to capture. For next two models of **Extension 1 and 2**, they show a **non-obvious decreasing trend** for valid loss which means they also learning something but not much in each iteration. Hence their test losses are very close to each other. The reason behind is because text-preprocessing removed a lot of redundant words which allows the model to learn more faster and converge.

For the worst model of **Baseline 1**, the **valid loss trends are unstable**, and **most of the time is increasing**. This is where **overfitting** occurs where the train loss is decreasing but the valid loss is increasing. Baseline 1 overfits earlier than other models which cause the worst test performance.

2.5 Quantitative Evaluation

	BLEU-4	**METEOR**	**Avg. % given items**	**Avg. extra items**
Baseline 1	0.0070	0.1120	30.32	22.18
Baseline 2	0.0098	0.1472	55.16	23.12
Extension 1	0.0158	0.1652	70.66	9.65
Extension 2	0.0097	0.1458	55.18	10.93

Matched with our expectation, baseline 1 has the **overall worst performance** on every metrics except the average extra items generated. Surprisingly, **Extension 1 model which ranked the second-best model has the best performance** on scoring the metrics with the most generated given items and least extra items. Highest score of BLEU-4 indicates to **close independent words** with the target sentences and highest METEOR score refers to the **most sensible sentence arrangement**. This is because the model focused the most to preprocess and **remove a lot of redundant words** which caused the model able to learn and form the recipe from the ingredients given, then arrange it into a proper sentence. For Extension 2, it seems like **even we introduced the penalty and compensation** mechanism to encourage the recipe generated contains as much ingredients given as possible, however it only scored the **second-best overall performance**.

In conclusion, **both extension models** improved and **outperformed the baseline 1 model**. However, the metrics scored by **baseline 2 is very close** to both extension models. This is potentially because of not filtering and preprocess the ingredients **very well** due to time and resource constraints. More time can be spent on studying and preprocess the ingredients into a proper list in future.

	BLEU-4	**METEOR**	**Avg. %given items**	**Avg. extra items**
Gold vs Sample	0.1889	0.5904	100	2

FIT5217 Assignment 2
Student Name: YI JIE NG
Student ID: 31158145

2.6 Qualitative Evaluation

Ingredients: 2 c sugar, 1/4 c lemon juice, 1 c water, 1/3 c orange juice, 8 c strawberries	
Baseline 1	in a large skillet , heat oil in a large skillet over medium-high heat . add o nion and cook until tender . add remaining ingredients and stir until well b lended . add remaining ingredients and mix well . pour into greased 9 " pi e plate . bake at 350 degrees for 30 minutes . <EOS>
Baseline 2	mix all ingredients together . in a large bowl , combine all ingredients . mi x well . pour into a greased 9 " x 9 " x 13 " baking pan . bake at 350 deg . f or 20 minutes or until golden brown . <EOS>
Extension 1	in a large saucepan combine water orange juice orange juice and water bri ng to a boil stirring constantly until thickened and bubbly stirring constant ly add lemon juice and lemon juice to taste and cook until thickened stirri ng constantly remove from heat and stir in lemon juice and lemon juice sti r in lemon juice and lemon juice and lemon juice and bring to a boil stirri ng constantly until the mixture thickens and thickens stir in the orange jui ce and lemon juice and lemon juice and cook stirring constantly until the mixture thickens stirring constantly remove from heat and stir in the oran ge juice and lemon juice and bring to a boil reduce heat to low and simme r for minutes or until the mixture thickens slightly thickened and bubbly s tirring constantly remove from heat and stir in the orange juice and lemon juice and lemon juice and bring to a boil stirring
Extension 2	combine all ingredients in a saucepan bring to a boil and simmer for minu tes or until the mixture is tender and the mixture is reached <EOS>

a. *Justification:*

Baseline 1 model used extra ingredients which can be said the worst result among all of the recipe in the table. Recipe generated by **Extension 1 model used a lot of ingredients given**, however it repeated the most compared to other models. Both models of **Baseline 2 and Extension 2 prefer to use general term** such as “mix and combine all ingredients” which is the smartest and most safe strategy. Hence there are both considered as acceptable result. However, **Baseline 2 model comparatively provided more details** which makes it slightly better than Extension 2 model.

The analysis above **strongly matches** and **proved the correlation** between quantitative metrics and qualitative behaviour. The best quantitative metrics of Extension 1 model matches the most ingredients given and the sentence output does make sense even though it is very long. Baseline 2 and Extension 1 models have both close quantitative metrics and qualitative result. Last but not least, Baseline 1 model is the only model which generated extra items, making it the lowest quantitative metrics score model.