FIT5217 - Assignment 2

Marks	Worth 100 marks, and 25% of all marks for the unit
Due Date	Week 12 Lecture Date, 11:55 PM
Extension	An extension could be granted under some circumstances. A special consideration application form must be submitted. Please refer to the university webpage on special consideration.
Lateness	For all assessment items handed in after the official due date, and without an agreed extension, a 10% penalty applies to the student's mark for each day after the due date (including weekends) for up to 7 days. Assessment items handed in after 7 days without special consideration will not be considered.
Authorship	This is an individual assessment. All work must be your own. All submissions will be placed through Turnitin. This makes plagiarism remarkably easy to identify for us.
Submission	4 files: a Jupyter notebook file, a pdf print of the Jupyter notebook, a csv file, and a 4 page pdf report need to be submitted. All answers in the report should be typed (no scanned documents). The name of the files must be `report_012345678.pdf`, `code_012345678.ipynb`, `code_012345678.pdf`, `generated_012345678.csv` where "`012345678`" is replaced by your own student ID. The report should be A4 size with standard margins and 11 point.

Table 1: Instructions for FIT5217 - Assignment 2

Introduction 🍕

Came back home after a long day, opened your fridge, you got a few ingredients in there, but have no clue how to cook something with them. You reach out for your 📞 and call your mom, but she does not answer your call. She had enough of you already! You reach out for your 📱, search for a cooking App, but surprisingly there is nothing in the App Store. [1] Well, you are a scientist, why not build one yourself? You could train a deep neural model to do the job for you. This is simply a Sequence-to-Sequence modelling task, mapping ingredients to a recipe.

Optional Literature Review 📝

No real research project could be done without a proper literature review. Here are some optional readings for you.

Paper: `Yinhong Liu, Yixuan Su, Ehsan Shareghi, Nigel Collier. Plug-and-Play Recipe Generation with Content Planning. In Proceedings of the 2022 Workshop on Natural Language Generation, Evaluation, and Metrics.`

Paper: `Chloé Kiddon, Luke Zettlemoyer, Yejin Choi. Globally Coherent Text Generation with Neural Checklist Models. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing.`

Paper: `Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, Yejin Choi. Simulating Action Dynamics with Neural Process Networks. In Proceedings of the 2018 International Conference on Learning Representations.`

Paper: `Abigail See, Peter J. Liu, Christopher D. Manning. Get To The Point: Summarization with Pointer-Generator Networks. In Proceedings of the 2017 Annual Meeting of the Association for Computational Linguistics.`

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.`

[1] Could someone actually check if there is an App for improvising food recipes? Share this in our Ed forum if you find anything.

Neural Chef Assistant 🤖 (Total 80+20 Marks)

The task is to train a deep neural model for recipe generation. The dataset we will use is from the Kiddon et al paper you (hopefully) reviewed above. Download the dataset from Moodle. You should use PyTorch for this part (need CE's approval if you want to use a different deep learning library). The test data should not be used for training. For each sample only use the ingredients and recipes, and ignore the titles, servings, categories. A sample from the data is demonstrated in Table 2.

We want you to first build two baseline models:

- Baseline 1: Sequence-to-Sequence model without attention [4 Marks]
- Baseline 2: Sequence-to-Sequence model with attention [5 Marks]

And then extend the Baseline that works better and build 2 other models [each worth 8 Marks] by choosing to do any of the followings (or a combination of them):

- Preprocessing data instead of using the original text (e.g., replacing all numbers with a unified token NUM, etc)
- Using pretrained embeddings (e.g., word2vec, GLoVe, etc)
- Separating or sharing the embedding matrices for encoder and decoder
- Stacking up more layers in encoder or decoder
- Using a different decoding strategy (i.e., beam search, top-k sampling, etc)
- Pretraining the encoder and decoder by training on Ingredients-to-Ingredients, or Recipes-to-Recipes. This is to train the encoder and decoder networks by reconstructing the same input at the output (a.k.a. auto-encoding), and re-using the pretrained encoder and decoder for Ingredients-to-Recipes.
- Any other substantial idea (should be a model design) from the papers reviewed in Part 1 (**Note: only those doing this will get the additional 20 Marks. You would need to provide 1-2 paragraphs of detailed description of what you have selected from those papers in Section 2.1. Use red font when including these details.**).

Note: The Jupyter notebook you submit should contain all the code and outputs needed to answer different sections of your report (see below). Make sure your code has clear markdowns (i.e., **Implementation of Baseline 1**, **Implementation of Baseline 2**, **Implementation of Extension 1**, **Implementation of Extension 2**) that are easy to navigate and locate which part of the notebook does what [5 Marks]. The discussion about the results should go into the corresponding sections of the pdf report.

Sections to include in your report: The structure of your report should have clear headings and be easy to navigate and read [5 Marks]. Avoid having several plots and tables if you could present them compactly in a single plot or table. Use the space wisely and be creative. Font size is 11, and page is A4. The report should cover the followings:

2.1 Model & Training Configurations. [5 Marks] For all your 4 models, use `LSTM` with `hidden_size` 256 as encoder and decoder, `teacher_forcing_ratio` 1, `Adam` as the optimizer with default hyperparameters, `dropout rate` 0.1, and set the `maximum length` to 150. You can choose the dimensionality of the word embeddings, number of training iterations, batch size, etc. But stick to what you choose in all your 4 models (do not change this across models). Report these model & training configurations along with training time (i.e., minutes) for each model explicitly in a table. You need to also report what decoding algorithm you are using. You also need to explicitly provide the detail of the hardware used for training the model (if used Google colab, just write Google Colab).

Title: chiles rellenos casserole
categories: vegetarian mexican main dish vegetables
servings: 10
ingredients: 2 cn whole green chili peppers 4 c milk 3 c sharp cheddar cheese 3/4 c all-purpose flour 4 green onions, sliced 1/4 ts salt 3 c shredded mozzarella cheese 2 cn green chili salsa 6 eggs
recipe: split chili peppers lengthwise and remove seeds and pith . spread chilies in a single layer in a greased 9x13-inch baking dish . sprinkle cheddar cheese , green onions , and 1-1/2 cups of the mozzarella cheese over chilies . in a bowl , beat eggs , milk , flour , and salt together until smooth . pour over chilies and cheese . bake in a 325 degrees oven for 50 minutes or until a knife inserted in custard comes out clean . meanwhile , mix salsa with the remaining 1-1/2 cups mozzarella cheese . sprinkle over casserole and return to oven for 10 minutes or until cheese melts . let stand for 5 minutes before serving .

Table 2: A sample from data.

2.2 Data Statistics. [5 Marks] Have a table with the statistics from training, dev, test sets you are using. This should report the statistics separately for ingredients, and recipes (i.e., have them as two separate rows). Statistics you need to report should include (but not limited to): number of samples, vocabulary size, min/max/average lengths of ingredients and recipes. Also explicitly mention if you have used the entire training data, or due to computational resource limitations used only a subset (if so, what percentage of the full training data you used?).

2.3 Data Preprocessing. [3 Marks] Mention if you have done any normalization or cleaning (i.e., removing tags, etc), or you just fed the data as-is into your model.

2.4 Analysis. [4 models× 3 marks each = 12 Marks] Include Training and Dev set loss plots (x-axis being the iterations, y-axis being the loss) for the 4 models. Discuss the plots: why one model performed better than the other, do you observe any sign of overfitting, etc. Try to use color and line styles to creatively and compactly present this in a single plot (i.e., you can use dashed and solid line styles to denote training and dev, and 4 different colors to represent different models). The axes need to have labels, and the plot needs a legend. Make sure axes ticks and labels are readable at 100% zoom.

2.5 Quantitative Evaluation. [4 models× 3 marks each = 12 Marks] Check Table 1 of Kiddon et al. They use 4 metrics to quantitatively evaluate the generated recipes from their models

on test set. Briefly describe these 4 metrics, report them for your 4 models on the test set. You can use NLTK to calculate BLEU-4 and METEOR, but need to implement the other two metrics (Avg. % given items, and Avg. extra items). Use a table similar to Table 3 to compactly present this. Discuss the results, how models compare, is something working surprisingly well? Have you outperformed any of the two baseline models? If not, what would you try in future?

Model	BLEU-4	METEOR	Avg. % given items	Avg. extra items
Baseline 1				
Baseline 2				
Extension 1				
Extension 2				

Table 3: Quantitative results on the recipe task for all models.

Have another table, similar to Table 4, where you use your implementation of metrics and calculate the 4 metrics on the Gold vs. Sample recipes provided in `metric_sample.txt`. For your convenience, in the provided `.txt` file, anything considered as ingredient is placed inside < `ingredient` >. This is for us to check if your metric calculation is correct or not.

	BLEU-4	METEOR	Avg. % given items	Avg. extra items
`Gold vs. Sample`				

Table 4: Metrics calculated on results on the sample recipes in `metric_sample.txt`.

2.6 Qualitative Evaluation. **[4 models× 2 marks each = 8 Marks]** Use a table similar to Table 5 where you report the generated output from each of your models for the following ingredient:

- Ingredients Sample 1: 2 c sugar, 1/4 c lemon juice, 1 c water, 1/3 c orange juice, 8 c strawberries

Discuss how models' outputs compare. Do you observe anything wrong (i.e., repetition, extra ingredients, etc)? Could you speculate why this happened, and what would you try differently in future to address this? Do you see any correlation between the quantitative metrics and the qualitative behavior of the models?

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
Generated Recipe ...	Generated Recipe ...	Generated Recipe ...	Generated Recipe ...

Table 5: Qualitative Samples from all models on the given ingredient list.

NOTE. All generated outputs from the 4 models should be reported in the corresponding column of the generated_012345678.csv file you submit for this assignment.