Deep Learning Project 2019

Neural Machine Translation Using Keras

Susan Wang
M20170548
Group 17

Contents

Task Definition

Explore the use of deep learning algorithms in creating machine translation models. This is a recent development which drastically simplifies the process of machine translation. Previously, machine translation has been done using statistical methods and requires complex rule-based logic. Neural machine translation uses quite simple deep learning models, and achieve comparable results with the statistical machine translation. It has been a major milestone in this field.

For this project, we have created a succession of deep learning models to translate French sentences to English sentences. We will compare the performance of each model, with the aim of creating a final model that achieves satisfactory performance for this task. This is a typical example of a Sequence to Sequence problem.

Dataset

The dataset was provided by Udacity, as a part of the Natural Language Processing nanodegree. Its attributes are as follows:

	French	***English***
Number of sentences	137,861	137,861
Vocabulary size	345	199
Total number of words	1,961,295	1,823,250
Maximum sentence length	21	15

The dataset has been pre-processed slightly, ie. converted to lowercase and punctuation tokenized with spaces.

Potential Challenges

Some of the main challenges in neural machine translations include:

- Long training time when working with large vocabulary
- Achieving adequate fluency
- Achieving correct translation when the vocabulary is out of context
- Weak performance when working with low frequency words
- Achieving correct translation for long sentences

As the dataset for this project contains a fairly small vocabulary as well as short to medium length sentences, it bypasses some of the challenges mentioned above.

Evaluation Measure

We will use the validation loss and accuracy to assess the model during training time, and then use the cumulative BLEU scores to evaluate the performance of each model. We will also examine a sample of translated sentences and compare the predictions with the target sentences in order to identify any inaccuracies in the translations.

Approaches

1. Preprocessing

The following pre-processing steps were applied to both French and English sentences

A. Tokenize words into unique integers using keras.preprocessing.text.Tokenizer´s fit_on_texts() method and texts_to_sequences() method
B. Pad each sentence to the maximum length of the French and English sentences, using keras.prerocessing.sequence.pad_sequences() method In this case they are all padded to 21 word vectors.
C. Reshape the target dataset to 3 dimensions, so that we can use keras[sparse categorical cross entropy function as the loss function.

2. Create Training and Test Sets

The dataset set was split into training and test sets in the ratio of 0.85 : 0.15. The training set is further split into training and validation during the model fitting process, where 80% of data was used for training and 20% was used for validation.

3. Designing Model Architecture and Parameters

We explored the following six different models, all using RNN, or more specifically, GRU. RNN has been designed for sequence prediction problems, while LSTM and GRU helps to overcome the issue of vanishing or exploding gradients. This is particularly useful for longer sequences. Here we decided to use GRU as they are simpler and faster to compute than LSTM, with comparable performances. Furthermore CuDNNGRU was used, to take advantage of the GPU on our machine.

1. Simple RNN
2. RNN with embedding
3. Bi-directional RNN
4. Encoder-decoder RNN
5. Combined model without Encoder-Decoder
6. Combined model with Encoder-Decoder

By building and analysing the performance of each of these models, we can see the effect of embedding, bi-directionality and using an encoder-decoder architecture on the accuracy of machine translation.

The target output has been formatted to have 3 dimensions, so that the loss function “sparse categorical crossentropy” can be used. We decided to use the Adam optimizer and measure accuracy as the metrics.

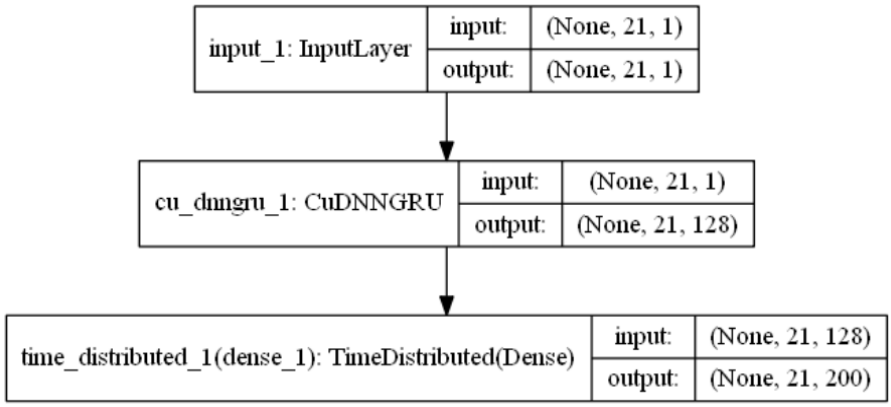
The following parameters have been chosen, after some brief experimentation. We attempted to use GridSearchCV in order to search for optimal hyper parameters but it was difficult to get around the error in keras caused by the dimensions required for the output layer.

num_rnn_units= 128
learning_rate = 0.01
test_batch_size = 512
embedding_dim = 128

Each model were trained for up to 30 epochs, with the exception of the simple encoder-decoder model, which was extended to 50 epochs. However, we also used early stopping criteria to monitor validation loss, and it will stop the training process when the validaton loss cease to decrease after 6 epochs.

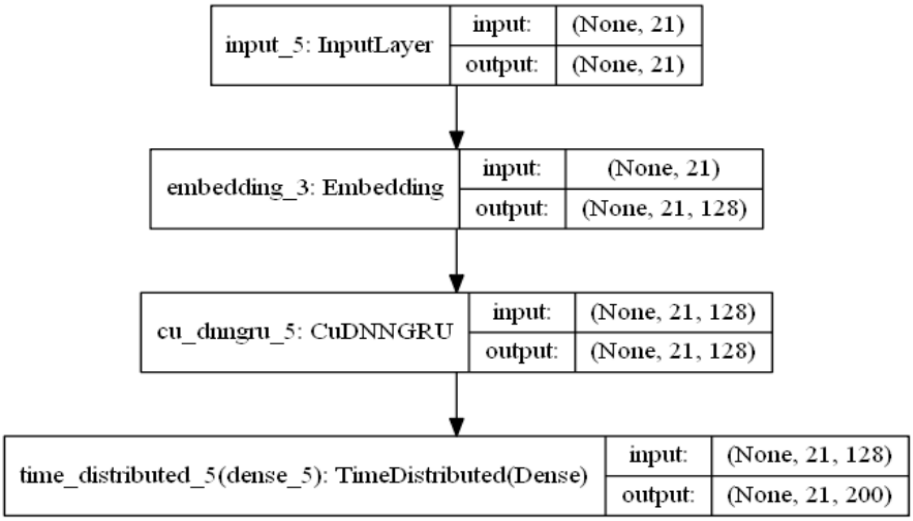
4. Build Models

a. Simple RNN



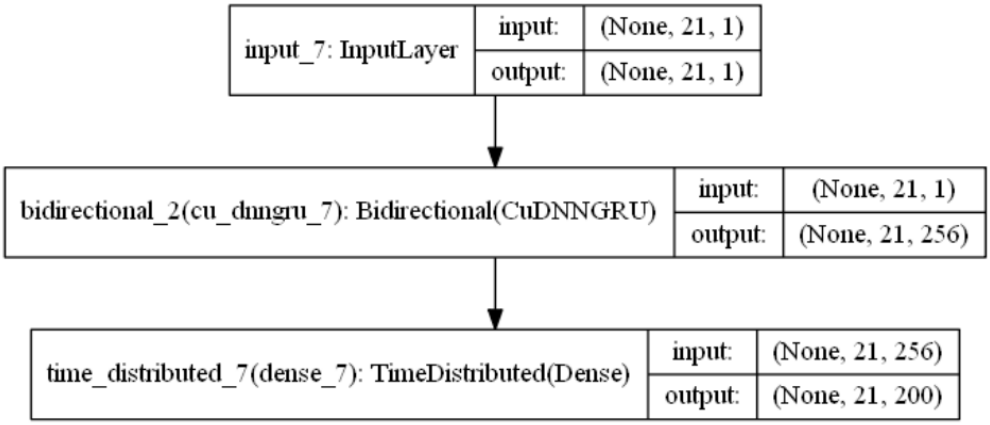
The input data is shaped to 3 dimensions, in order to work with Keras GRU layer.
The GRU layer Is set with “return_sequences” to True, so that it passes a 3D output to TimeDistributed layer. This means each of the 128 GRU units return a sequence of 21 outputs, one for each time step in the input data.
The TimeDistributed layer then applies the same Dense layer to the GRU output one time step at a time, such that the output layer only needs one connection to each GRU unit. The output layer is configured to be the output vocabulary size (+1 for padding), with “softmax” activation layer, so that it can predict the likelihood of each output vocabulary for each of the 21 timestep.

b. RNN with embedding



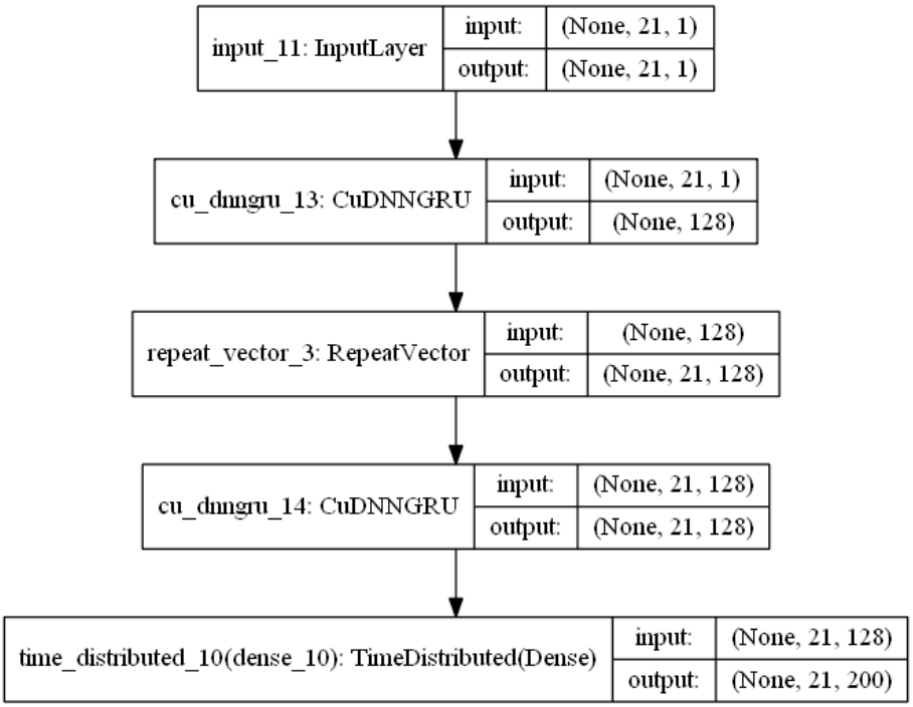
This model extends the Simple RNN model by including a custom embedding layer. To do so, the input dataset need to be shaped to have 2 dimensions instead of 3. The embedding layer transforms each input integer into a embedding vector of 128 dimensions, and passes the result to the subsequent GRU layer. The remainder of the architecture is the same as for the Simple RNN model.

c. Bi-directional RNN



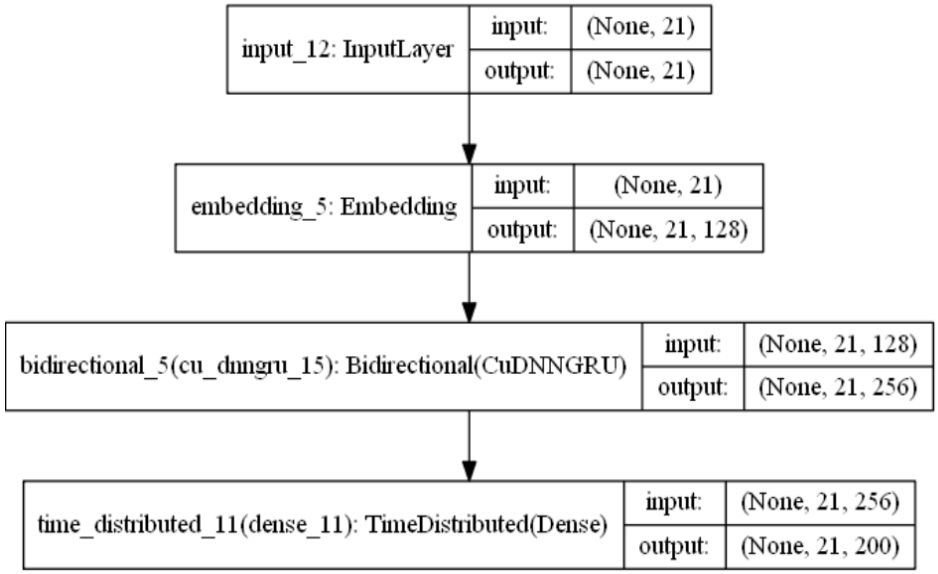
This model extends the Simple RNN model by making the GRU layer bi-directional. This effectively doubles the number of GRU units, as well as taking into account both the forward and reverse dependency of the input sequence.

d. Encoder-Decoder RNN



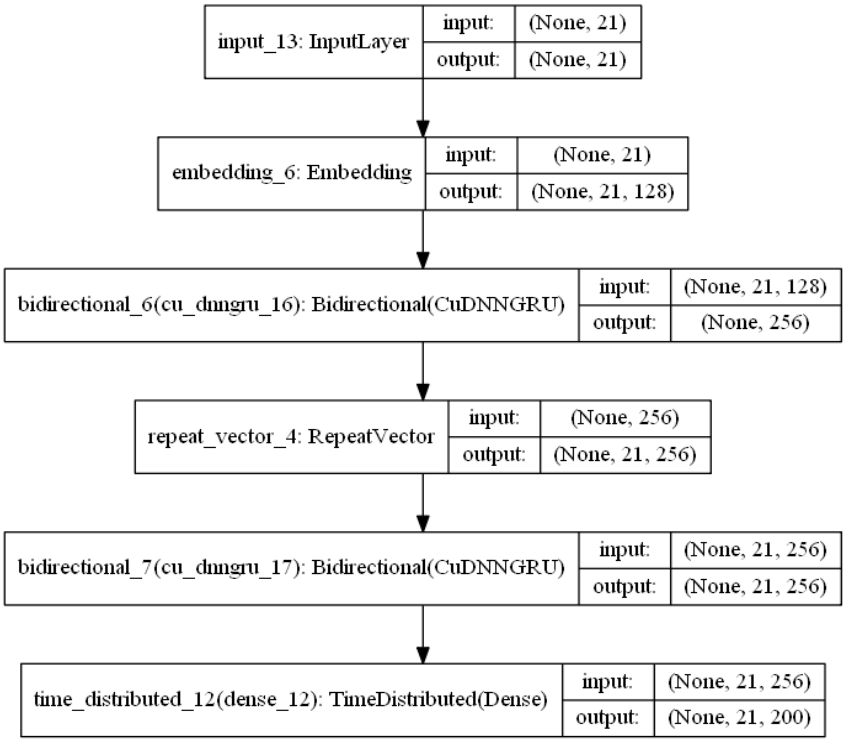
Here we extend the Simple RNN model by incorporating the popular encoder-decoder mechanism that is frequently used for sequence to sequence tasks. The first GRU layer acts as an encoder, creating a single vector representation of the 21 timesteps from the input layer. This information is passed to the Repeat Vector layer, which repeats the encoded input vector 21 times and then passes onto the decoder, also made of GRU units. The subsequent layers behave as per previous models. Since the encoder and decoders are decoupled, theoretically the decoder layer only needs to process 15 timesteps, since that is the maximum length of the output sequence. (However we have kept it as 21 units for now..)

e. Combined model 1 - without encoder-decoder



In order to see the effect of encoder-decoder architecture on the overall model, we decided to configure two separate final models, one with encoder-decoder and one without. For this first model, we have combined the embedding layer with the bi-directional GRU layer, but without the encoder-decoder set up.

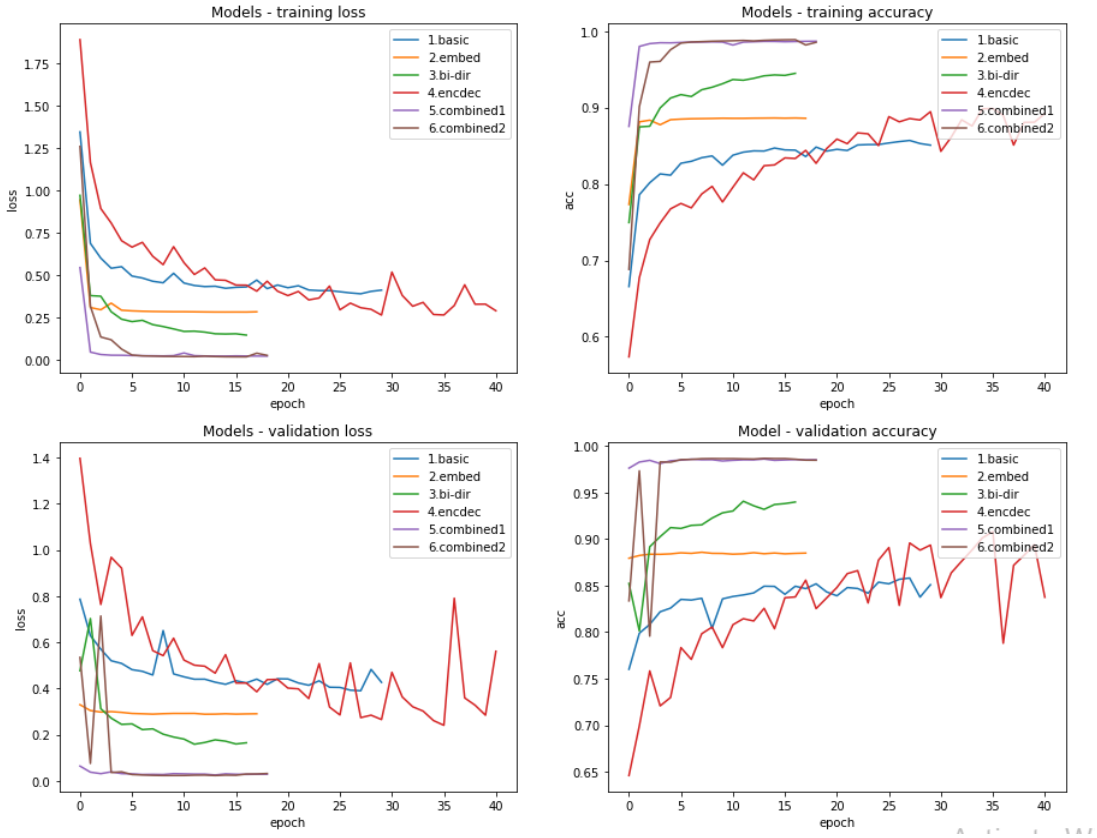
f. Combined model 2 - with encoder-decoder



This is a combined model with the encoder-decoder architecture. By doing so, we can compare the effect of encoder-decoder configuration on the final model performance.

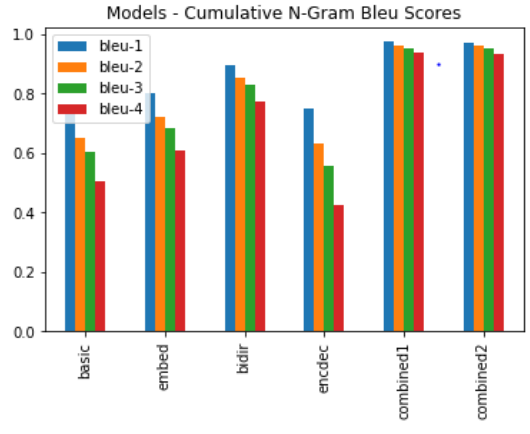
5. Evaluate Models

a. Model metrics



The graphs clearly show a consistent order of performance for the 6 models, where the basic RNN model and the encoder-decoder model are the worst performing models. The model with embedding has a moderate performance, which is surpassed by the simple, bi-directional RNN model. The combined models achieved similar performance, ie. The encoder-decoder architecture did not make a significant difference.

b. Cumulative BLEU



	bleu-1	bleu-2	bleu-3	bleu-4
basic	0.748662	0.651351	0.603070	0.507528
embed	0.799664	0.719834	0.684355	0.607672
bidir	0.896982	0.853186	0.828067	0.773097
encdec	0.750845	0.630631	0.556733	0.424679
combined1	0.975068	0.962787	0.954511	0.937992
combined2	0.973331	0.960213	0.951683	0.934166

c. Visual comparison of sample sentences

	target	predicted (basic)	predicted (embed)	predicted (bidir)	predicted (encdec)	predicted (combined1)	predicted (combined2)
0	[[california, is, sometimes, dry, during, may, and, it, is, sometimes, wonderful, in, february]]	[california, is, sometimes, busy, during, august, but, it, is, usually, in, in, summer]	[california, is, sometimes, dry, during, may, but, it, is, sometimes, in, in, autumn]	[california, is, sometimes, dry, during, may, and, it, is, sometimes, wonderful, in, october]	[california, is, sometimes, dry, during, may, and, it, is, never, mild, in, october]	[california, is, sometimes, dry, during, may, and, it, is, sometimes, wonderful, in, february]	[california, is, sometimes, dry, during, may, and, it, is, sometimes, wonderful, in, february]
1	[[california, is, never, hot, during, autumn, but, it, is, never, dry, in, march]]	[california, is, never, hot, during, autumn, but, it, is, usually, hot, in, summer]	[california, is, never, hot, during, summer, but, it, is, sometimes, hot, in, march]	[california, is, never, warm, during, autumn, but, it, is, never, dry, in, march]	[california, is, sometimes, warm, during, fall, but, it, is, never, rainy, in, winter]	[california, is, never, warm, during, autumn, but, it, is, never, dry, in, march]	[california, is, never, hot, during, autumn, but, it, is, never, dry, in, march]
2	[[the, united, states, is, sometimes, rainy, during, january, but, it, is, mild, in, may]]	[the, united, states, is, sometimes, quiet, during, january, but, it, is, usually, in, september]	[the, united, states, is, sometimes, rainy, during, january, but, it, is, mild, in, may]	[the, united, states, is, sometimes, rainy, during, january, but, it, is, wet, in, may]	[the, united, states, is, sometimes, snowy, during, january, but, it, is, beautiful, in, summer]	[the, united, states, is, sometimes, rainy, during, january, but, it, is, mild, in, may]	[the, united, states, is, sometimes, rainy, during, january, but, it, is, mild, in, may]
3	[[they, like, apples, oranges, and, pears]]	[they, dislike, bananas, lemons, and, lemons]	[they, like, mangoes, mangoes, and, lemons]	[they, like, apples, oranges, and, oranges]	[they, like, apples, apples, and, pears]	[they, like, apples, oranges, and, pears]	[they, like, apples, oranges, and, pears]
4	[[he, dislikes, mangoes, and, strawberries]]	[he, dislikes, oranges, and, and]	[he, dislikes, oranges, grapes, grapes]	[he, dislikes, grapes, and, pears]	[he, dislikes, mangoes, and, pears]	[he, dislikes, strawberries, and, strawberries]	[he, dislikes, mangoes, and, strawberries]
5	[[the, grapefruit, is, her, favorite, fruit, but, the, lemon, is, your, favorite]]	[the, grapefruit, is, her, least, favorite, but, the, lemon, is, my, favorite]	[the, grapefruit, is, her, least, favorite, but, the, lemon, is, your, favorite]	[the, grapefruit, is, his, favorite, fruit, but, the, lemon, is, your, favorite]	[the, grapefruit, is, her, favorite, fruit, but, the, grapefruit, is, his, favorite, favorite]	[the, grapefruit, is, his, favorite, fruit, but, the, lemon, is, your, favorite]	[the, grapefruit, is, her, favorite, fruit, but, the, lemon, is, your, favorite]
6	[[india, is, sometimes, pleasant, during, autumn, but, it, is, usually, nice, in, march]]	[india, is, never, dry, during, november, and, it, is, never, in, in]	[india, is, never, hot, during, august, but, it, is, never, in, in, autumn]	[india, is, sometimes, pleasant, during, autumn, but, it, is, usually, nice, in, march]	[india, is, sometimes, warm, during, autumn, but, it, is, usually, cold, in, winter]	[india, is, sometimes, pleasant, during, autumn, but, it, is, usually, pleasant, in, march]	[india, is, sometimes, pleasant, during, autumn, but, it, is, usually, pleasant, in, march]
7	[[your, most, loved, fruit, is, the, apple, but, his, most, loved, is, the, peach]]	[your, least, loved, fruit, is, the, strawberry, but, his, most, loved, is, the, strawberry]	[your, most, loved, fruit, is, the, mango, but, my, most, loved, is, the, mango]	[your, most, loved, fruit, is, the, apple, but, his, most, loved, is, the, strawberry]	[my, most, favorite, fruit, is, the, pear, but, her, her, favorite, is, the]	[your, most, loved, fruit, is, the, apple, but, her, most, loved, is, the, peach]	[your, most, loved, fruit, is, the, apple, but, her, most, loved, is, the, peach]
8	[[the, united, states, is, cold, during, winter, and, it, is, sometimes, relaxing, in, june]]	[the, united, states, is, cold, during, summer, and, it, is, usually, hot, in, november]	[the, united, states, is, chilly, during, summer, but, it, is, sometimes, hot, in, march]	[the, united, states, is, chilly, during, winter, and, it, is, sometimes, wonderful, in, june]	[the, united, states, is, nice, during, winter, and, it, is, never, rainy, in, june]	[the, united, states, is, chilly, during, winter, and, it, is, sometimes, relaxing, in, june]	[the, united, states, is, chilly, during, winter, and, it, is, sometimes, relaxing, in, june]
9	[[i, like, grapefruit, apples, and, mangoes]]	[when, think, grapefruit, peaches, and, grapes]	[i, like, grapefruit, peaches, and, lemons]	[i, like, grapefruit, peaches, and, grapes]	[you, like, grapefruit, peaches, and, mangoes]	[i, like, grapefruit, apples, and, mangoes]	[i, like, grapefruit, apples, and, mangoes]

This table shows predicted output from a sample of 10 sentences from the test dataset. Differences are highlighted in yellow and are discussed in the following page.

d. Model Results Comparison

Model 1 – Simple RNN
The program correctly predicted many words, but sometimes get keywords wrong, and sometimes repeats the same word multiple times, and sometimes miss words and ends the sentence in illogical places. The BLEU-1 score is 0.749, suggesting the accuracy of unigram words, but BLEU-4 of 0.508 suggested that it only gets 4 consecutive words correctly roughly half the number of times.

Model 2 - Simple Embedding
The embedded model improved upon the basic RNN model by getting more keywords correct, for instance, it correctly predicted the words "dry", "rainy" and "mild", but it still made many mistakes in the prediction of fruits. This is possibly due to insufficient data to correctly differentiate between the fruits in the embedding. The same mistakes also occurred in the prediction of months. There are still words being repeated, such as "in" occurring twice immediately after each other in some sentences.

Model 3 - Simple Bi-directional RNN
The bi-directional model is a significant improvement on the previous models, in that there are no more duplicated words, so that the translated sentence reads better. Although there are still some errors in the individual word translations, especially when it comes to fruits and months, the model does a good job in correctly translating consecutive words, as shown by BLEU-4 score of 0.773, which is a considerable improvement over the embedding model of 0.608 and basic model of 0.508

Model 4 - Simple encoder-decoder RNN
The encoder-decoder model encodes the input sentence into a single context vector and then decodes it into a different language. The result shows that the output sentence reads well and does not have duplicate words, however some of the translation are incorrect. We decided to let this model run for up to 50 epochs, to give it time to improve the accuracy. However the end result is still quite poor, in fact it is sometimes worse than the result of the basic model. This is likely due to the fact that all the information is compressed in a single vector representation, which is insufficient to represent the entire sentence. It appears that the encoder decoder method needs to be combined with other approaches in order to maximise its performance. Or perhaps be constructed in a different way, such that the decoder input takes in both the context vector and the encoder input, so that it can be more specific in the decoding.

Model 5 – Combined model 1 – without encoder-decoder RNN
The combined model does a fairly good job in translating the test sentences. It achieved BLEU-4 score of 0.938, suggesting it is able to get consecutive words correct majority of the time. Comparing the predicted output with the target output above, we can see that there are only six differences in the 10 sentences. Three of those differences are related to warm instead of hot, chilly instead of cold, and pleasant instead of nice. All of which means the same thing so are insignificant. Two of the six differences are due to gender pronoun prediction, which is always difficult to get right. The last error was the prediction of strawberry instead of mango, which is possibly due to inadequate number of samples for this particular context. All in all, the model performed very well, considering it is still quite a simple model. This is likely due to the fact the dataset only contained a small set of vocabulary for both English and French, and at the same time there are proportionally large amount of training data

Model 6 – Combined model 6 – with encoder-decoder RNN
This appears to be the best performing model, when looking at the sample translations of the test dataset in page 8. It achieved similar performance to the combined model 1, which did not have the encoder-decoder architecture. In the sample prediction, there are only 3 difference between the predicted sentences and the target sentences. Two of the differrences minor (pleasant instead of nice, and chilly instead of cold). The third difference is due to the gender pronoun, which is quite difficult to predict correctly when the sentences are short and did not have a clear context. The BLEU-4 score of 0.934 suggest the model is able to correctly predict consecutive words correctly majority of the time.

6. Conclusions

It has been an interesting exercise creating machine translation models using Keras. The chosen dataset contains only a small vocabulary, but with sufficient training samples, which made this project easier to master. We have explored different architectures and evaluated the results in detail, and in doing so gained further appreciation of the effect of different elements in the translation outcome.

By combining various components into the keras RNN model, we were able to achieve a cumulative BLEU-4 of 0.93, which is quite a good outcome. It is perhaps surprising that the encoder-decoder architecture did not perform as well as we had expected, and perhaps we could explore a different way of constructing the encoder decoder model so that the decoder takes in both the initial input sentence as well as the encoder context vector, so that the context can be used in conjunction with individual words in the input sentence.

Below are the extensions made to the Udacity project:

1. Instead of english to french, changed to french to english… so we can better evaluate the fluency of the translation
2. Split into training, validation and test set.
3. Implemented a combined model without encoder-decoder
4. Implemented BLEU score to evaluate the translations
5. Implemented callbacks and early stopping criteria
6. Draw the models
7. Plot the results of accuracy
8. Plot the results of BLEU
9. Evaluated the translation provided by each model

7. Future Work

It would be worthwhile to continue this topic with the following extensions:

1. Cross validation and hyper-parameter tuning
2. Attention
3. Transformer
4. Use a more complex dataset