Haiku generation with RoBERTa

1. Team member

Yusuke Miyashita 30510694

2. Introduction

Haiku is a type of poetry originated in Japan and it evokes images of nature. Haiku consists of 3 lines of 5, 7, 5 words. The goal of the application is to generate a Haiku based on the input words from the user.
Some research on generating Haiku Has been done using LSTM, a deep learning technique. However, I have not come across any research on Haiku using a transformer. Based on the online resources and literature review, a transformer-based machine learning technique for NLP called RoBERTa will be used for this application. RoBERTa is suited for this application as the model is trained by predicting multiple masked words in a sentence. In other words, this model is suited to generate a sentence(Haiku) from a few words (user input). I have found a pre-trained RoBERTa on English text, which we will use as the baseline for our work.

___ New ___ ___ ___ cute little ___ at ___ ___

On New Year's Day a cute little pilgrim at the gate

Figure showing the 4 input words and the expected Haiku from the model

3. Related project

LSTM

In the past, some research on Haiku generation has been done by LSTM. However, I have not come across any research about generating Haiku with a transformer based model. For a mask prediction task such as Haiku, I expect the transformer model to perform better in generating Haiku just as the transformer model out performed the LSTM model's score on various NLP benchmarks in "Attention Is All you Need".
In addition, transformers are faster to train than LSTM as the model can process multiple words in a sentence rather than one word at a time.

BERT

BERT (Bidirectional encoder representation from transformer) is a machine learning algorithm published by google in 2018. One of the advantages of BERT is that Language model like LSTM only uses left or right context but BERT learns context from all positions of words in a sentence. It has achieved state of the art results in varieties of NLP benchmarks. In the BERT's training process, it receives a sentence

with a mask and predicts it. This can be extended into a training with multiple masks and can be directly used as a Haiku generator

4. Dataset

http://haikuguy.com/issa/search.php

Extraction of data

A python library called “Beautifulsoup” is used to extract English haiku from the website. The extracted haiku is then processed before being saved in csv format. The processing of the Haiku includes removing some special characters such as [! , ‘ , , , ?, --] which are not necessary for the generation of Haiku. “ \n” is also removed to make Haiku become 1 line sentence instead of 3 lines so that the model can read Haiku as a sentence. This is one of the compromises of the project. Similarly, some
 are removed to make sure there is only 1
 between words in a sentence as RoBERTa tokenises space as well, meaning treating space as 1 word.

Data breakdown

Extracted Haiku	Train Haiku	Validate Haiku	Test Haiku
11370	9096	1137	1137

5. Approach

Masking algorithm

Some words in the extracted Haiku need to be masked in order to fine-tune RoBERTa. The aim is to mask most words in a sentence, except for a few words. This can be done by implementing an algorithm to mask each word in a sentence with a certain probability. For example, if a sentence/Haiku consists of 10 words, and you want to leave 3 words without masking, we can set the probability of masking each word to be 70% to have 7 words masked.

```
def random_mask(sent):
  words = sent.split()
  for i in range(len(words)):
    random_value = random.randint(0, 10)
    #0,1,2,3,4,5,6,7,8,9. 40% of chance to get value greater than 5
    if random_value > 5:
      words[i] = '<mask>'
  # this can remove the unnecessary space between words in haiku e.g. "an  apple"
  sent = ' '.join(words)
  #sent = sent.join("</s>")
  return sent
```

Tokenization

Tokenization is the process of translating a meaningful piece of data such as string to a number that can be processed mathematically by RoBERTa.

masked haiku

```
down to two leaves <mask> lonely morning <mask> spring frost
```

label Haiku

```
down to two leaves the lonely morning glory spring frost
```

tokenized masked haiku

```
tensor([[    0,  3955,     7,    80,  3607, 50264, 20100,   662, 50264,  2428,
         18082,     2]])
```

tokenized label haiku

```
tensor([[    0,  3955,     7,    80,  3607,     5, 20100,   662, 12594,  2428,
         18082,     2]])
```

Figure showing the examples of mask, label and tokenization

Model - RoBERTa

A Robustly Optimised BERT Pretraining Approach (RoBERTa) is a deep learning algorithm for NLP tasks. RoBERTa is implemented by modifying key hyperparameters in BERT and training with much larger mini-batches and learning rates. This allows RoBERTa to improve on the masked language modeling objective compared with BERT.

Some elements of RoBERTa are

Positional encoding

Positional encoding allows the model to learn relative positioning of a word in the sentence. Tokenized Haiku is embedded with positional encoding before fed to the model

Attention

Attention is used in the transformer encoder block. Attention is a neural network architecture to identify which parts of the input sequences are relevant to each word of the output.

Loss function

Cross entropy loss is used as the loss function in the model. The model predicts the masked word in haiku and compares it to the label to calculate the loss.

$$L_{\text{cross-entropy}}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i} y_i \log(\hat{y}_i)$$

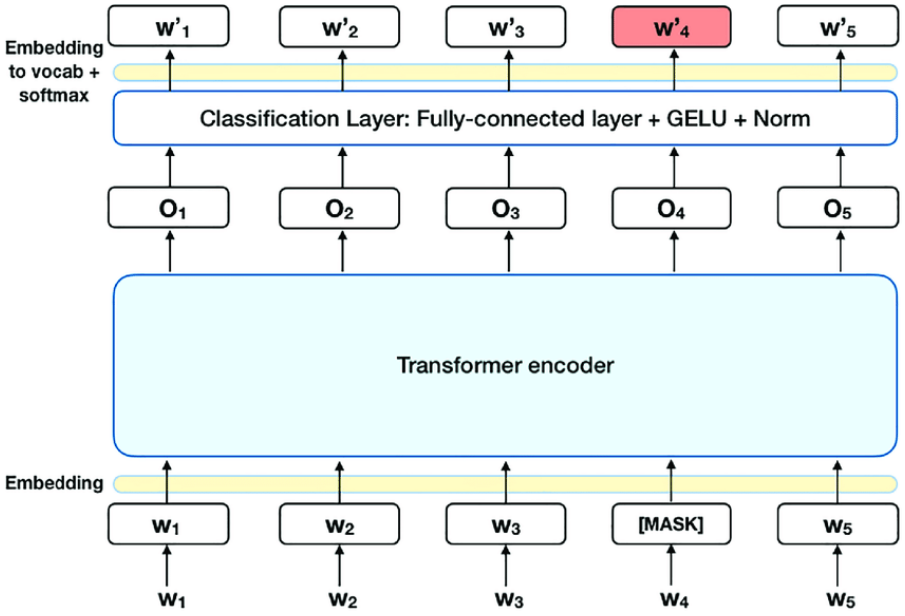


Figure: architecture of BERT

Optimiser

AdamW algorithm

---

input : γ(lr), β_1, β_2(betas), θ_0(params), $f(\theta)$(objective), ϵ (epsilon)
λ(weight decay), $amsgrad$

initialize : $m_0 \leftarrow 0$ (first moment), $v_0 \leftarrow 0$ (second moment), $\widehat{v_0}^{max} \leftarrow 0$

---

for $t = 1$ **to** $\ldots$ **do**

$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$

$\theta_t \leftarrow \theta_{t-1} - \gamma\lambda\theta_{t-1}$

$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$

$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$

$\widehat{m_t} \leftarrow m_t / \left(1-\beta_1^t\right)$

$\widehat{v_t} \leftarrow v_t / \left(1-\beta_2^t\right)$

if $amsgrad$

$\widehat{v_t}^{max} \leftarrow \max(\widehat{v_t}^{max}, \widehat{v_t})$

$\theta_t \leftarrow \theta_{t-1} - \gamma\widehat{m_t} / \left(\sqrt{\widehat{v_t}^{max}} + \epsilon\right)$

else

$\theta_t \leftarrow \theta_{t-1} - \gamma\widehat{m_t} / \left(\sqrt{\widehat{v_t}} + \epsilon\right)$

---

return θ_t

Training algorithm

```
def train(input, label):
    input = input.to(device)
    label = label.to(device)
    outputs = model(input, labels=label)
    loss = outputs.loss
    loss.backward()
    optimizer.step()
    lr_scheduler.step()
    optimizer.zero_grad()
    progress_bar.update(1)
    return loss
```

6. Analysis

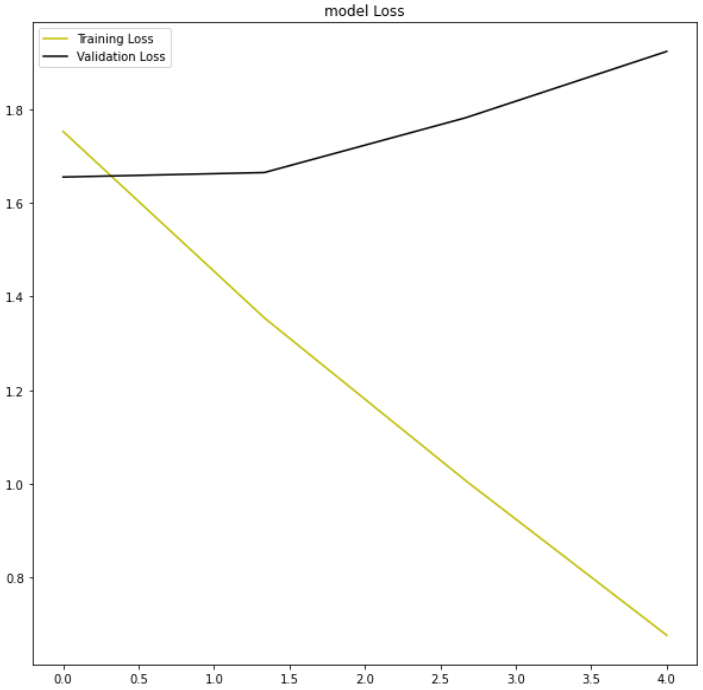
Training of 4 epochs took approximately 40mins

hyper parameters

Learning rate	Lr scheduler	epochs	Batch size
5e-5	Linear, warmup_step=0	4	1

By checking training and validation error, all the hyperparameters are chosen through trial and error. There is not much significance in the initial learning rate as it is adjusted by the lr scheduler for each batch.

Loss plot



In the plot, the training plot decreases while the validation loss increases. This is strange as this has been happening since epoch 1. A possible reason for this issue is because the dataset haiku has been translated from Japanese to English and has some grammatical problems. Alternatively, RoBERTa is pretrained on a dataset which has a different sentence structure than Haiku, and this difference may affect how a loss is calculated, causing some increase in validation error.

Loss on Test Haiku

Model with least training loss is used

```
test loss: 1.9754798412322998
```

Visualization of predictions

Model with least training loss is used

```
Original Sentence:   ___ New ___ ___ ___ cute little ___ at ___ ___
label:  on New Years Day a cute little pilgrim at the gate


Mask  1 Guesses :  ['The', 'A', 'Brand', 'Meet', 'In']
Mask  2 Guesses :  ['York', 'Yorkers', 'Orleans', 'Yorker', 'Jersey']
Mask  3 Guesses :  ['has', 'finds', 'found', 'launches', 'have']
Mask  4 Guesses :  ['a', 'the', 'this', 'these', "'s"]
Mask  5 Guesses :  ['girl', 'puppies', 'boy', 'puppy', 'baby']
Mask  6 Guesses :  ['the', 'her', 'their', 'his', 'a']
Mask  7 Guesses :  ['party', 'wedding', 'zoo', 'event', 'mall']
Before train: Best guess for fill in the blank:   The York has a girl the party


Mask  1 Guesses :  ['on', 'in', 'by', 'a', 'the']
Mask  2 Guesses :  ['Years', 'Year', 'Day', 'Month', 'Japan']
Mask  3 Guesses :  ['presents', 'gifts', 'Day', 'pine', 'charm']
Mask  4 Guesses :  ['a', 'this', 'the', 'such', 'digs']
Mask  5 Guesses :  ['house', 'pine', 'gate', 'tree', 'hut']
Mask  6 Guesses :  ['the', 'my', 'your', 'his', 'our']
Mask  7 Guesses :  ['gate', 'window', 'door', 'gates', 'inn']


After train: Best guess for fill in the blank:   on Years presents a house the gate
```

As seen in the visualisation, the trained model has better guess than the pretrained model.

7. Conclusion

The pipeline of fine-tuning pretrained RoBERTa model and generation of Haiku has been done and achieved reasonable output though many compromisations. In visualising the prediction of the masked words, the fine-tuned model produced better prediction than the pretrained model.
The scope of this application is quite narrow due to the time constraint. However, there are many things that can be done in future to improve the model of Haiku generation.

8. Future work

Restrict Tokenizer to produce multiple tokens from one word

RoBERTa inherits its tokenizer algorithm from BERT and the tokenizer can break a word into sub-word, resulting in multiple tokens Thus, the length of tokenized masked Haiku and its tokenized label do not match. The model requires both masked and label to have the same length to make the prediction. As a result, only 50 % of the extracted Haiku have been used for fine-tuning the model.

New metric to measure the accuracy of the prediction

By visualising the predictions, it becomes clear that the fine-tuned model generates more accurate predictions than the pretrained model, even when the validation loss increases. Other methods might be needed to verify this result and accurately calculate the validation loss. Alternatively, more datasets need to be screened by visualisation to clarify the fine-tuned performs better than the pre-trained model.

Generate Japanese Haiku

The dataset Haiku is originally written in Japanese and then translated to English. As stated in the movie Paterson, “Poetry in translation is like taking a shower with a raincoat on.” (Jarmusch, 2016). It would be amazing to train RoBERTa with Japanese Haiku. This should be a low-hanging fruit as it would only require a Japanese dataset and the model pre-trained in Japanese.

9. References

AdamW - Pytorch documentation
https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html

Attention Is All You Need
https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
https://arxiv.org/abs/1810.04805

Development of Automatic Haiku Generator Using LSTM
https://www.jstage.jst.go.jp/article/pjsai/JSAI2018/0/JSAI2018_1B2OS11b01/_pdf/-char/ja

Generation and selection of Haiku Poems Using Deep Learning
https://www.jstage.jst.go.jp/article/jjsai/34/4/34_467/_pdf/-char/ja

Jarmusch, J. (Director). 2016. Paterson

RoBERTa: A Robustly Optimized BERT Pretraining Approach
https://arxiv.org/abs/1907.11692