

Homework 7 - Generative Models - Spring 2025

Essay Question: *While recent advances in generative models appear to realistically mimic human responses to prompts, these models are still far from being able to pass the Turing test.* Write a paragraph discussing your opinion regarding the validity of this assertion and provide explicit examples supporting your viewpoint.

Notebook Assignment 1: Text summarization is the process of distilling essential information from a piece of text, thereby creating an abridged version of the text that retains its core meaning. There are two types of text summarization: extractive, which selects and stitches together existing sentences, and abstractive, which generates new sentences to convey the main points.

The BART (Bidirectional and Auto-Regressive Transformer) is a transformer based model pre-trained on a denoising autoencoder objective. BART has shown strong performance in a number of NLP tasks that include "abstractive" text summarization. It is pre-trained on a de-noising autoencoder target, which allows it to generate a coherent and contextually relevant summaries. Essentially we are treating the full text as a "noisy" sequence and the BART model is finding the noise-free version of the text.

BART is built upon the Transformer architecture. This model uses self-attention mechanisms to capture contextual relationships between words in a sequence. Before feeding text into the BART model, the input text is tokenized into smaller units called tokens. The BART model extends the transformer architecture with a specific pre-training objective. It is trained to denoise documents corrupted by various techniques like shuffling and removing sentences.

BART is pre-trained using a denoising autoencoder objective. Given a corrupted input sequence X' , the model is trained to reconstruct the original sequence X . The objective is to minimize the reconstruction loss

$$L(X, X') = - \sum_{i=1}^n \log \Pr(X_i | X'_1, X'_2, \dots, X'_{i-1}, X'_{i+1}, \dots, X'_n)$$

During the fine tuning phase for summarization, BART is trained on summarization specific datasets to fine-tune its ability to generate abstractive summaries. The objective during summarization involves generating a summary Y from the input sequence X . BART is fine-tuned to minimize the negative log-likelihood of the target summary given the input:

$$L_{\text{summary}}(X, Y) = - \sum_{j=1}^m \log \Pr(Y_j | X, Y_1, Y_2, \dots, Y_{j-1})$$

In this demo

1. The datasets have been preprocessed and stored as TF dataset objects in the zip file. The saved dataset objects use a Batch size of 8. Load both data set objects as `train_ds` and `val_ds`. Print out the first batch from the training dataset. Examine the raw SMS text and target summary from this first batch. Comment on how well you feel the target summarizes the actual message for each SMS text.
2. Use the following to load a preprocessor for the BART model, setting the maximum encoder/decoder sequence lengths to 512 and 128, respectively.

```
preprocess = keras_hub.models.BartSeq2SeqLMPreprocessor.from_preset (
    "bart_base_en",
    encoder_sequence_length = MAX_ENCODER_SEQUENCE_LENGTH,
    decoder_sequence_length = MAX_DECODER_SEQUENCE_LENGTH,
)
```

Then use the following to load the BART model.

```
bart_lm = keras_hub.models.BartSeq2SeqLM.from_preset (
    "bart_base_en", preprocessor = preprocessor
)
```

Print the model summary to identify the number of trainable parameters.

3. Instantiate an AdamW optimizer initialized as follows

```
optimizer = keras.optimizers.AdamW(
    learning_rate = 5e-5,
    weight_decay = 0.01,
    epsilon = 1e-6,
    global_clipnorm = 1.0, #gradient clipping
)
#exclude layer norm and bias terms from weight decay
optimizer.exclude_from_weight_decay(var_names = ["bias"])
optimizer.exclude_from_weight_decay(var_names = ["gamma"])
optimizer.exclude_from_weight_decay(var_names = ["beta"])
```

Compile the model with this optimizer and a sparse categorical crossentropy loss function.

```
loss = keras.losses.SparseCategoricalCrossentropy(from_logits=True)
bart_lm.compile( optimizer = optimizer,
    loss=loss, weighted_metrics=["accuracy"],
)
```

OPTIONAL: Train your model for at least 2 epochs and save the model after each training epoch. This TODO task is optional because the model takes 3-5 hours per epoch to train on a machine with a single CPU.

4. I've provided a link on the course website to a model that has been trained for two epochs. Note that the file is large (1.6GB). Reload this model into your notebook and use it to generate text summaries for 5 samples in the validation dataset, `val_ds`. Print the text dialogue, generated summary, and target summary for each sample. Comment on the accuracy of the generated summaries.