**Big\_Patent Summarization**

**6/19/2023**

As advised, I started reading about different fine-tuned models available for text summarization. I have come across several language models and figured out these 20 state-of-the-art models which can generate summaries from patent documents.

* BART
* GPT-J
* Longformer
* T5
* RoBERTa
* Pegasus
* ConvBERT
* CTRL
* FlauBERT
* FNet
* GPT NeoX
* MarianMT
* MPNet
* mT5
* Open-LLAMA
* ProphetNet
* RoCBert
* RoFormer
* T5v1.1
* XLM-RoBERTa
* XLNet

I classified the above models into 4 different categories based on the architecture. I am going to work on two to three models from each category to generate the summary.

**Transformer-based models:**

Transformer-based models are neural networks that are composed of a stack of self-attention layers. Self-attention is a mechanism that allows the model to learn the relationships between different words in the input text. This is done by comparing each word in the input text to all of the other words in the text, and then assigning a weight to each word based on how relevant it is to the other words. The weights are then used to compute a new representation of the input text, which is then passed to the next layer of the model.

Transformer-based models are particularly well-suited for natural language processing tasks because they are able to learn long-range dependencies between words. This is because the self-attention mechanism allows the model to compare words that are far apart in the input text. This is important for tasks such as text summarization, where the model needs to understand the overall meaning of the text in order to generate a summary.

Transformer-based model architecture

A picture containing text, diagram, screenshot, plan

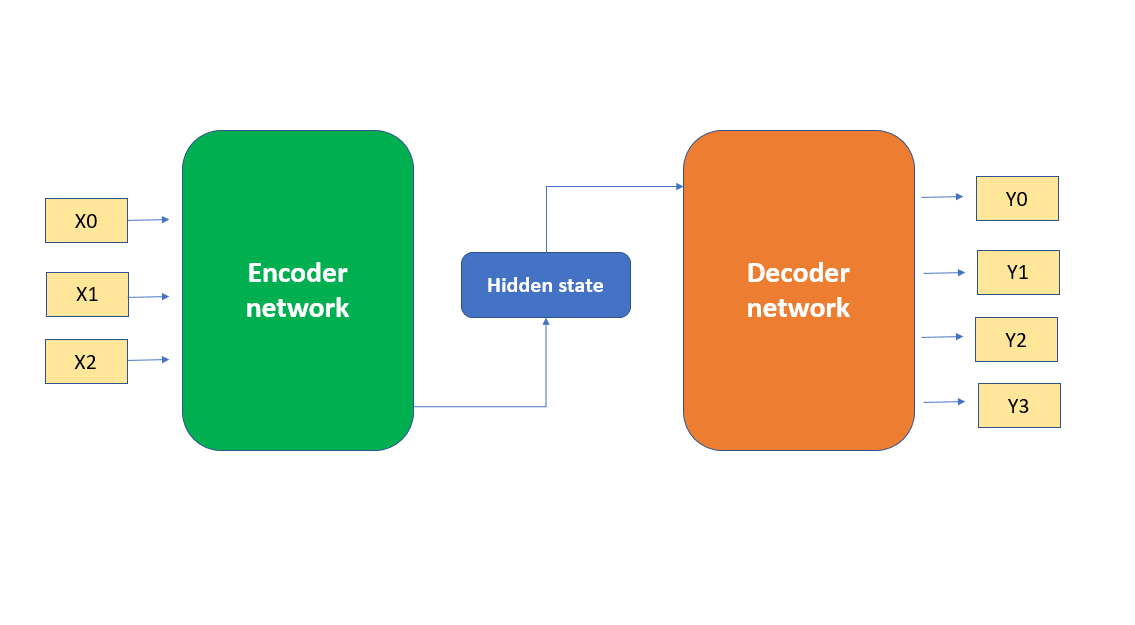
Description automatically generated

Some of the most popular transformer-based models for generating summaries include BART, T5, RoBERTa, and GPT-J.

**Encoder-decoder models:**

Encoder-decoder models are neural networks that are composed of two parts: an encoder and a decoder. The encoder reads the input text and produces a sequence of hidden states. The decoder then takes these hidden states as input and generates the output text.

Encoder-decoder model architecture



The encoder is typically a transformer-based model, while the decoder can be a variety of different models. The most common type of decoder is a recurrent neural network (RNN). RNNs are able to process text one word at a time, which makes them well-suited for tasks such as text summarization, where the model needs to generate the output text one word at a time.

Some of the most popular encoder-decoder models for generating summaries include Pegasus, MarianMT, and MPNet.

**Bidirectional models:**

Bidirectional models are able to read text in both directions. This means that they are able to capture the context of the text more effectively than models that can only read text in one direction. This is because bidirectional models are able to see how words are related to words that come before and after them.

A picture containing diagram, line, font, circle

Description automatically generated

Bidirectional models are a popular choice for natural language processing tasks that involve understanding the meaning of text. This is because the ability to capture the context of the text is essential for understanding the meaning of complex sentences.

Some of the most popular bidirectional models for generating summaries include CTRL, FlauBERT, and FNet.

**Multi-task models:**

Multi-task models are trained on multiple tasks. This allows them to learn more generalizable features. This is because the model is able to learn features that are useful for multiple tasks. This can be beneficial for tasks such as text summarization, where the model needs to learn a variety of different features in order to generate a good summary.

A picture containing screenshot, text, diagram, design

Description automatically generated

Some of the most popular multi-task models for generating summaries include GPT NeoX, Open-LLAMA, and ProphetNet.

I picked three models from each category, and they are as below.

Transformer-based Model:

* BART ( Completed)
* T5 (Completed)
* RoBERTa
* GPT-J

Encoder-Decoder Model:

* BigBird-Pegasus (Completed)
* MarianMT
* MPNet

Bidirectional Model:

* CTRL
* FlauBERT
* FNet

Multi-Task Model:

* GPT NeoX
* Open-LLAMA
* ProphetNet

I also generated Abstract + Claim summary for the Pegasus Model.

**Bigbird+ Pegasus Model:**

BigBird-Pegasus is a sparse-attention based transformer which extends Transformer based models, such as BERT, to much longer sequences1. In addition to sparse attention, BigBird also applies global attention as well as random attention to the input sequence. It can handle sequences up to a length of 4096 at a much lower compute cost compared to BERT. It has achieved state-of-the-art results on various tasks involving very long sequences such as long documents summarization and question-answering with long contexts.

BigBird-Pegasus is a large language model (LLM) chatbot developed by Google AI. It is a combination of the BigBird and Pegasus models. BigBird is a sparse-attention based transformer which extends Transformer based models, such as BERT to much longer sequences. In addition to sparse attention, BigBird also applies global attention as well as random attention to the input sequence. Pegasus is a generative pre-trained transformer model that is specifically designed for summarization tasks.

BigBird-Pegasus was trained on a massive dataset of text and code, including books, articles, and code. This gives it the ability to generate summaries that are both comprehensive and informative. It is also able to generate summaries of technical or scientific texts.

**Model:**

<https://colab.research.google.com/drive/1ck1Ugpr_s7XHJmm9M41lnOGSohjNdPgp?usp=sharing>

**Output:**

<https://docs.google.com/spreadsheets/d/1wZliSEreaCnL6h21r5d_p31PiOXDApwm/edit?usp=drive_link&ouid=113406982075106512493&rtpof=true&sd=true>

**Bart Model:**

BART is a denoising autoencoder pre-trained on a massive dataset of text and code. It is a powerful and versatile model that can be used for a variety of natural language processing tasks, including summarization, translation, and question answering. BART is able to generate text that is both informative and fluent, ansd it is relatively easy to use and fine-tuned for specific tasks. However, it is a large model, which can be computationally expensive to train and run.

BART (Bidirectional and Auto-Regressive Transformers) is a sequence-to-sequence model trained as a denoising autoencoder for natural language tasks. It uses a standard seq2seq/machine translation architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT). The pretraining task involves randomly shuffling the order of the original sentences and a novel in-filling scheme, where spans of text are replaced with a single mask token. BART is particularly effective when fine-tuned for text generation but also works well for comprehension tasks.

**Model:**

<https://colab.research.google.com/drive/1kRLwUXr1SihKH_E4ggONvpX7A51hYKs-#scrollTo=XmyuyOydP2d9>

**Output:**

<https://docs.google.com/spreadsheets/d/1zRE6IEF9XY175EF0DsV3_MMMJv8qv1CO/edit?usp=sharing&ouid=113406982075106512493&rtpof=true&sd=true>

**GPT\_J:**

GPT-J (Generative Pre-trained Transformer 3) is a large language model chatbot developed by EleutherAI. It is a 6 billion parameter model trained on a massive dataset of text and code. GPT-J is able to generate text, translate languages, write different kinds of creative content, and answer your questions in an informative way.

It generally follows the GPT-2 architecture with the only major difference being the so-called parallel decoders: instead of placing the feed-forward multilayer perceptron after the masked multi-head attention, they are computed in parallel in order to achieve higher throughput with distributed training1. GPT-J performs very similarly to similarly-sized OpenAI’s GPT-3 versions on various zero-shot downstream tasks and can even outperform it on code generation tasks1. The newest version, GPT-J-6B, is a language model based on a dataset called The Pile.

**Model:**

<https://colab.research.google.com/drive/1K8dw994AxdIU6QJAriBgr9TAQfI0zccS?usp=sharing>