# Natural Language Generation

# Textbook

Denis Rothman - Transformers for Natural Language Processing\_ Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more-Packt Publishing Ltd

# Text Generation with OpenAI GPT-2

## [code](https://github.com/PacktPublishing/Transformers-for-Natural-Language-Processing.git)

OpenAI wanted to create a task-agnostic model. They began to train transformer models on raw data instead of relying on labeled data by specialists. Labeling data is time-consuming and considerably slows down the transformer's training process.

The first step was to start with unsupervised training in a transformer model. Then, only to fine-tune the model's supervised learning.

Humans learn a language and then apply that knowledge to a wide range of NLP tasks through transfer learning. The core concept of a T5 model is to find an abstract model that can do things like us.

A trained model will behave like a person who learned a language. It will understand what it can and learn from input data.

Hugging face [model](https://huggingface.co/models)

Hugging face [dataset](https://huggingface.co/datasets)

Hugging face [metrics](https://huggingface.co/metrics)

# Text summarization with T5

## installation

* Pip install transformers==4.0.0
* Pip install sentencepiece==0.1.94
* *pip install torch==1.4.0*
* conda install pytorch torchvision torchaudio cudatoolkit=11.0 -c pytorch

## code

import torch

import json

# transformers==4.26.1

# sentencepiece==0.1.97

# torch==1.13.1

# conda install pytorch torchvision torchaudio cudatoolkit=11.0 -c pytorch

from transformers import T5Tokenizer, T5ForConditionalGeneration, T5Config

model = T5ForConditionalGeneration.from\_pretrained('t5-large')

tokenizer = T5Tokenizer.from\_pretrained('t5-large')

device = torch.device('cpu')

display\_architecture=True

def summarize(text,ml):

  preprocess\_text = text.strip().replace("\n","")

  t5\_prepared\_Text = "summarize: "+preprocess\_text

  print ("Preprocessed and prepared text: \n", t5\_prepared\_Text)

  tokenized\_text = tokenizer.encode(t5\_prepared\_Text, return\_tensors="pt").to(device)

  # summmarize

  summary\_ids = model.generate(tokenized\_text,

                                      num\_beams=4,

                                      no\_repeat\_ngram\_size=2,

                                      min\_length=30,

                                      max\_length=ml,

                                      early\_stopping=True)

  output = tokenizer.decode(summary\_ids[0], skip\_special\_tokens=True)

  return output

#Bill of Rights,V

text ="""Hello, my name is Abhijith M and I am excited to introduce myself to you.

I am a Senior Software Engineer with 3 year and 9 months of experience in

Artificial Intelligence, Machine Learning, Deep Learning and Computer Vision projects.

Now, I am working in Zerone Consulting.

I had completed my B.Tech Computer Science and Engineering under Kerala Technological University in 2019.

I have experience in R and Python.

Throughout my career, I have had the opportunity to work with some amazing teams and clients,

and have been able to learn and grow a great deal as a professional.

I pride myself on being a hardworking, dedicated, and enthusiastic individual

who is always eager to take on new challenges and opportunities.

I am passionate about play chess and enjoy using my free time to reading.

"""

print("Number of characters:",len(text))

summary=summarize(text,100)

print ("\n\nSummarized text: \n",summary)

while True:

    text = input("Enter text to continue: ")

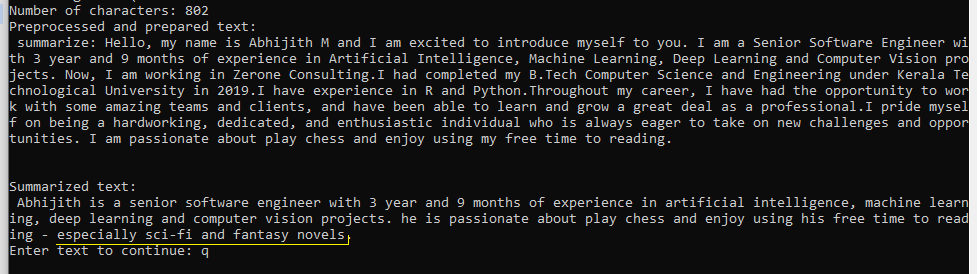
    if text == "q":

       break

    print("Number of characters:",len(text))

    summary=summarize(text,100)

    print ("\n\nSummarized text: \n",summary)



# Semantic Role Labeling

* !git clone https://github.com/primeqa/primeqa.git
* !pip install faiss-cpu==1.7.2 faiss-gpu==1.7.2
* !pip install datasets==2.0.0
* !pip install scikit-learn
* !pip install --force transformers==4.17.0
* !pip install numpy==1.21.6

import os

import json

os.chdir("primeqa")

os.listdir()

from primeqa.components.reader.extractive import ExtractiveReader

reader = ExtractiveReader(model="PrimeQA/nq\_tydi\_sq1-reader-xlmr\_large-20221110")

reader.load()

context = [["""Hello, my name is Abhijith M and I am excited to introduce myself to you. I am a Senior Software Engineer with 3 year and 9 months of experience in Artificial Intelligence, Machine Learning, Deep Learning and Computer Vision projects. Now, I am working in Zerone Consulting. I had completed my B.Tech Computer Science and Engineering under Kerala Technological University in 2019. I have experience in R and Python. Throughout my career, I have had the opportunity to work with some amazing teams and clients, and have been able to learn and grow a great deal as a professional. I pride myself on being a hardworking, dedicated, and enthusiastic individual who is always eager to take on new challenges and opportunities. I am passionate about play chess and enjoy using my free time to reading."""]]

question = ["What is his name?"]

answers = reader.predict(question,context)

print(json.dumps(answers, indent=4))

print(answers["0"][0]["span\_answer\_text"])

Abhijith M