



# North South University

## CSE499B: Senior Design Project II

### Updated System Design

**“Autonote: Transformative meeting summarization and highlighting points based on NLP”**

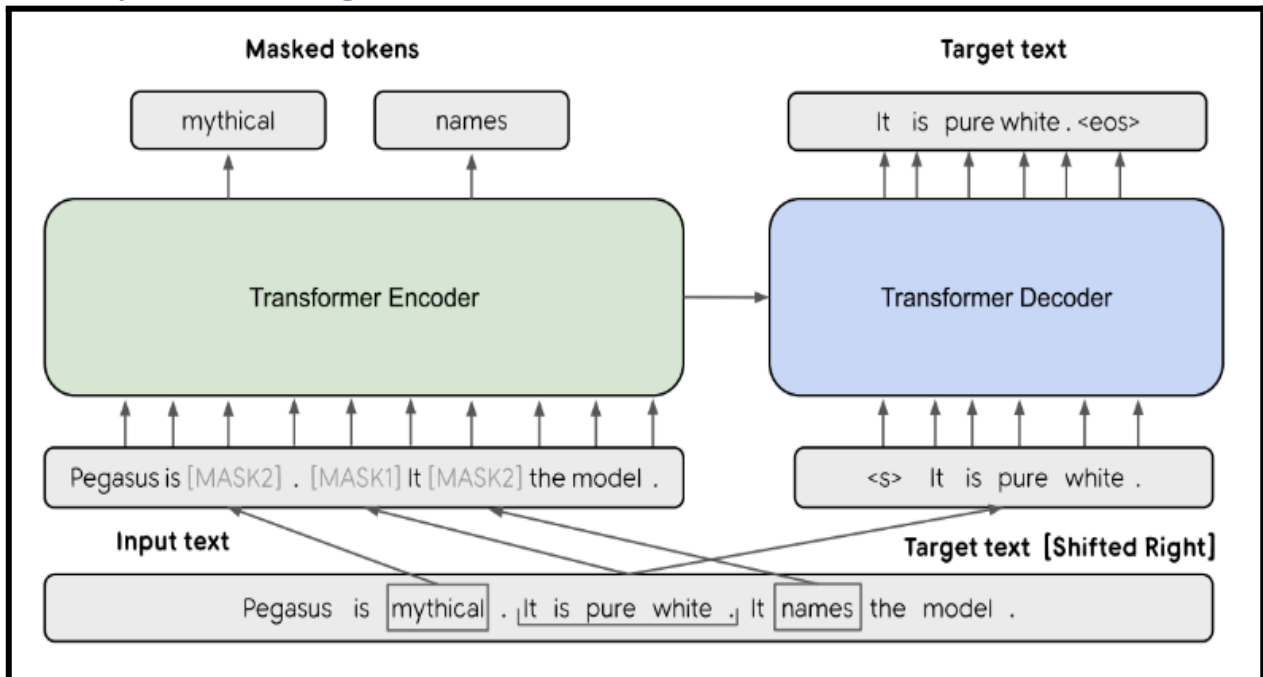
Group Members:

Serial	Name	NSU ID	Section
1	Sheikh Mohammed Wali Ullah	2021186042	10
2	Md. Saiyem Raiyan	2012468042	10
3	Zobaer Ahammod Zamil	2021796042	10
4	Samia Sultana	2014048042	29

# Updated System Design

## CSE499A

### Previous System Design:



### Audio Conversion (Audio to Transcript)

Figure 1: System Design of the model using Pegasus-CNN-Dailymail model

**Model Use:** *google/pegasus-cnn\_dailymail*

**Dataset Use:** *samsum*

The previous system generates a summary from the meeting transcript. *pegasus-cnn\_dailymail* model uses the “*samsum*” dataset where dialogue with summaries exists in that dataset. The hugging API generates the summary also from the transcript. The meeting audio generated a transcript conversion. Then, input the transcript and make the summary of the transcript. The data was collected from a YouTube meeting transcript from audio voice to caption of the subtitle. Therefore, it converted into audio to text. After that, segmentation is the part that divides the data into segments that can be easily analyzed, processed, and converted into text. We fine-tuned the previously selected *Pegasus-CNN-Dailymail* model.

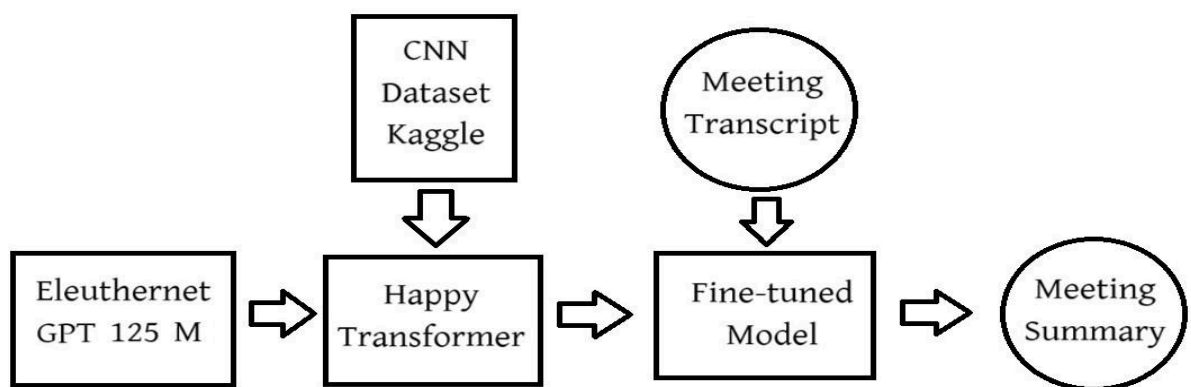
According to these criteria, the model is fine-tuned and trained up to **15 epochs** and Tested with **12 fields** ( paper of Pegasus model) related data. So, in exploratory data analysis, we find a short summary with dataset validation. Besides, it handles problematic transcripts and irrelevant summaries. Our model type is a transformer that learns context and thus meaning by tracking relationships in sequential data like the words in this sentence and evaluating the metrics in rouge scores. To get the best and optimal result, we fine-tuned the model with changing parameters and penalty scores and stratified it with a validation set. Here, we used Stochastic gradient descent (SGD) and normalization. Finally, evaluate the model with performance on validation and testing datasets.

### ***What is the problems with pegasus model:***

According to our pegasus model, the summary cannot work well. Sometimes, it gives a short summary of the pegasus model, which is good. However, in most cases, it cannot give the meaningful summary that ChatGPT or hugging face API gives. 12 field testing from the pegasus model is good in some cases, but with a huge and long transcript, it cannot give a meaningful summary. So, the model cannot work efficiently. Thus, the summary cannot work well in this format according to the pegasus model. So, by analyzing the new model, we found the **EleutherAI/gpt-neo-2.7 B** model, a transformer model designed using EleutherAI's replication of the GPT-3 architecture. GPT-Neo refers to the class of models, while 2.7B represents the number of parameters of this particular pre-trained model. The EleutherAI model works more efficiently than pegasus cnn-dailymail.

## **CSE499B**

***Shift with a new model*** ➡ ***EleutherAI/gpt-neo-2.7B***



***Model Use (Load): EleutherAI/gpt-neo-2.7B***

***(<https://huggingface.co/EleutherAI/gpt-neo-2.7B>)***

***Model Use (Load+Train): EleutherAI/gpt-neo-125M***

***(<https://huggingface.co/EleutherAI/gpt-neo-125m>)***

***Transformer Model: Happy Transformer (<https://happytransformer.com/text-generation/>)***

***Dataset Use: CNN-DailyMail News Text Summarization***

***(<https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail>)***

The system generates a summary from the meeting transcript. **EleutherAI/gpt-neo** model uses the “**CNN-DailyMail News Text Summarization**” dataset where dialogue with summaries exists in that dataset. Here, we load the model by a transformer model called “**Happy Transformer**”. We load the model as a generator. It means load with HappyGenerator() function [Code in below]. Firstly, we load this gpt-neo model with 2.7 Billion parameters. It was loaded successfully but unfortunately due to it's huge size it couldn't be able to train (cuda memory broke down with 2x T4-GPU and 29 GB RAM).

However, for this problem we again try with this model but this time with 125 Million Parameters for training with cnn dataset and HappyGenerator. Then we tried to train with 1 epoch. It was successfully loaded again and trained [Code snippets and screenshot below].

We tested the train set first with a customized dataset.

***How we customize dataset for prompt:*** There are 3 columns in the datasets. Given below:

**train.csv** (1.26 GB)

Detail    Compact    Column

### About this file

287,113 unique articles were provided for train.

<b>A id</b>	<b>A article</b>	<b>A highlights</b>
heximal formatted SHA1 hash of the url where the story was retrieved from	a string containing the body of the news article	a string containing the summary of the article
<b>287113</b> unique values	<b>284005</b> unique values	<b>282197</b> unique values
0001d1afc246a7964130f43ae940af6bc6c57f01	By . Associated Press . PUBLISHED: . 14:11 EST, 25 October 2013 .   .	Bishop John Folda, of North Dakota, is taking time off after being

We merged it in just 1 column. The input text: in the first then summary: like **Text**  $\diamond$ . **Summary:**  $\diamond$

So that, the result will be a summary from the given text. Therefore, the training process with 1 epoch was done.

**train25000.csv** (109.51 MB)

Detail Compact Column

```
A text
24983
unique values

(CNN) -- Ralph Mata
was an internal
affairs lieutenant
for the Miami-Dade
Police Department,
working...

A drunk driver who
killed a young woman
in a head-on crash
while checking his
mobile phone has
been ...
```

The actual dataset size is huge (almost 1.6 GB) which break the training System. For this reason we split the dataset with 3 segments. Firstly, split with 2500 cells, then 25000 cells, then 50000 cells, the 100000 cells. But again it breaks for 50000 and 100000 cell train sets. Therefore, we worked with 25000 cell dataset that we split before. Then we trained. The dataset now like this with one column. *Text*  $\diamond$ . *Summary*:  $\diamond$

## One Sample:

Beirut (CNN) -- Syria carried out an airstrike on a refugee camp in northern Lebanon Saturday, killing nine Syrians and wounding nine more, a Lebanese state-run news agency reported. The strike centered on a Syrian refugee camp located near the Syrian border between the towns of Baalbeck and Aarsal in the Bekaa Valley, the National News Agency said. The Red Cross took the casualties to Universal Hospital in Baalbek. Saturday's strike was not the first by the Syrian government, which has accused rebels of smuggling arms and supplies across the border. On March 18, two Syrian jets fired three rockets that hit empty buildings near Aarsal. At the time, a U.S. State Department spokeswoman called the use of fighter jets to fire rockets into Lebanon a "significant escalation." U.N. commissioner wants to probe into whether Syrian rebels executed soldiers. Also in March, the U.N. Security Council voiced "grave concern over repeated incidents of cross-border fire which caused death and injury among the Lebanese population, incursions, abductions and arms trafficking across the Lebanese-Syrian border, as well as other border violations." The declaration followed a briefing by officials on how the conflict in Syria has spilled into Lebanon. More than 600,000 Syrians have fled to neighboring Lebanon, a country of about 4 million people, according to a U.N. estimate. But the Lebanese government puts the total at more than 1 million. Whatever the true figure, there is no dispute that the influx has destabilized the area and heightened tensions. The attack comes as the Syrian conflict is mired in a third year of unrest, which started in March 2011 when President Bashar al-Assad cracked down on peaceful protesters. Since then, it has evolved into a civil war that has killed more than 100,000 and transformed more than 1 million others into refugees, according to the Red Cross. Read more: U.N. inspectors heading to Syria to probe chemical weapons report. CNN's Nick Paton Walsh reported this story from Beirut, and Tom Watkins wrote it in Atlanta. CNN's Hamdi Alkhashali and Yousuf Basil contributed to this report. Summary: Airstrike kills nine Syrians in refugee camp, state media reports.

Syria has fired into Lebanon before .

The government has accused rebels of smuggling arms across the border with Lebanon .

**Training Results:** Invalid text [Output given below]

```
args2 = GENSSettings(min_length=70, max_length=100)
result = happy_gen.generate_text("A thief nicknamed the 'Black Widow of Facebook' - who drugged and robbed men after she lured them into bed has
print(result.text)
```

Setting `pad token id` to `eos token id`:50256 for open-end generation.

[illegible]

## ***What is the problems with gpt neo and happy transformer model:***

According to *EleutherAI/gpt-neo model and happy transformer* the summary can not be able to work. This gives redundant output, the same token repeated again and again [screenshot above]. Anyhow it can not generate any good summary. For this major problem we again shift with a new model called “**Phi-2**” from microsoft.

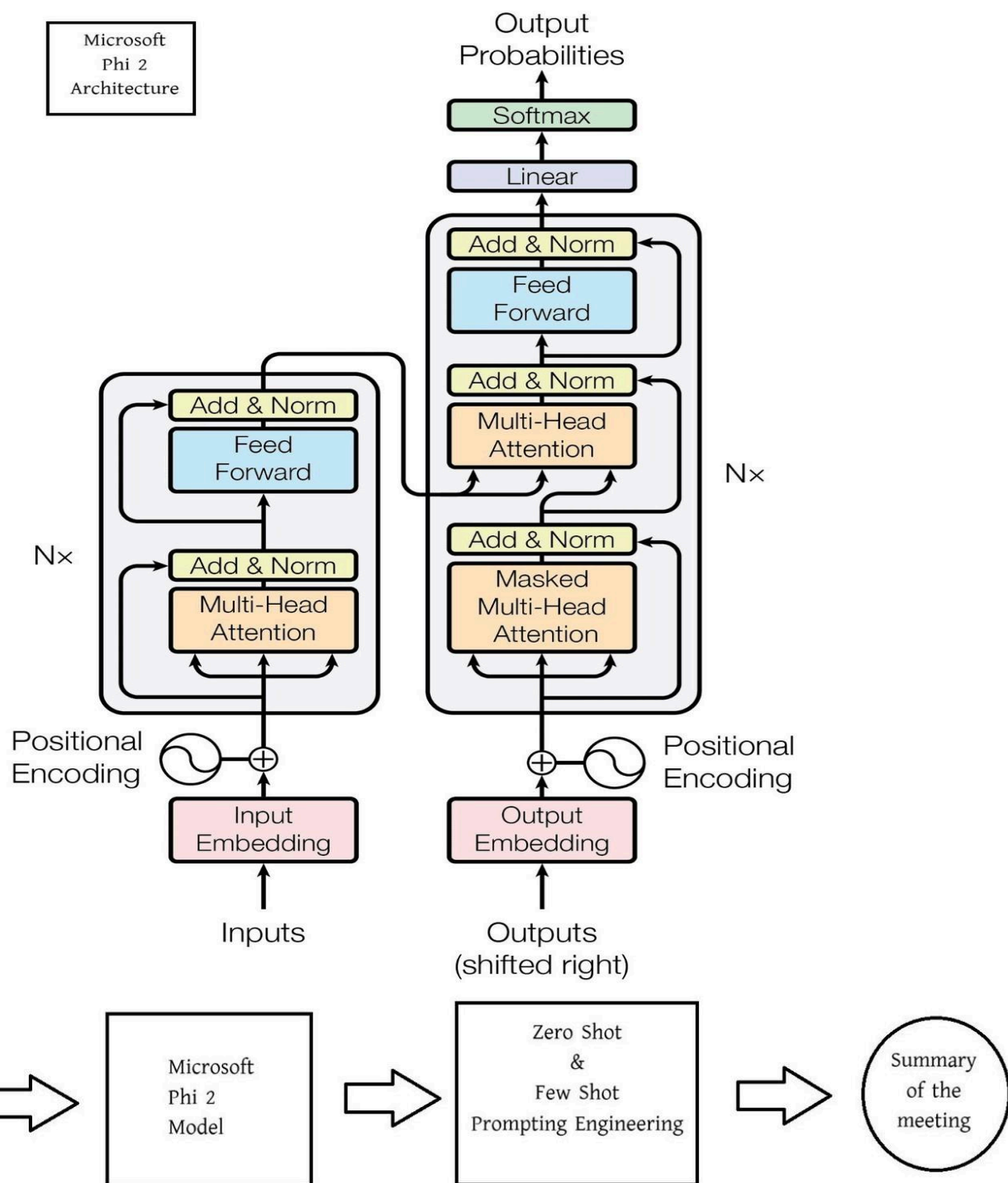
## **Workings with Phi-2**

***Shift with a new model*** ➡ ***microsoft/phi-2***

(<https://huggingface.co/microsoft/phi-2>)

***Dataset Use: CNN-DailyMail News Text Summarization***

(<https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail>)





The system generates a summary from the meeting transcript. “**Phi-2**” model uses the “**CNN-DailyMail News Text Summarization**” dataset where dialogue with summaries exists in that dataset. The meeting audio generated a transcript conversion. Then, input the transcript and make the summary of the transcript. Firstly, we import a text generation model from the Hugging Face Transformers library and initialize it with a specified model called "microsoft/phi-2". The pipeline function sets up a text generation pipeline with the specified model. Then generates a text summary based on the provided prompt using a text generation model. The summary is presented using Markdown format. Then initializes a tokenizer for the "phi-2" model with a causal language model ("AutoModelForCausalLM") for the "microsoft/phi-2" model, with automatic specification of torch data type and device mapping. Note that the model is pretrain. So, we didn't train for it. Finally, we generate a short summary of the given prompt using the initialized language model. It utilizes the tokenizer to encode the prompt and the model to generate the summary based on the encoded prompt. The summary is then decoded and printed.

### Sample Output:

```
prompt = "Give me a short summary of the following text. Asana and Avir are twins who just passed HSC and want to pursue higher educ
with torch.no_grad():
    token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
    output_ids = model.generate(
        token_ids.to(model.device),
        max_new_tokens=512,
        do_sample=True,
        temperature = 0.3
    )

output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
print(output)
```

The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's `attention\_mask` to obtain reliable results.  
Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

Output: Asana and Avir, twins who recently completed high school, want to pursue higher education together in computer science. Asana seeks her brother's help to convince their father to let her join Avir in the same university under the same program. However, their father rejects Asana's choice of computer science and urges her to consider business or arts programs instead. Asana turns to her brother for support, but he remains silent along with their mother. In contrast, Avir receives approval to study his desired field without any opposition.  
<|endoftext|>

Most importantly, we tried the *zero shot technique* here to generate a summary. We'll try *few shot techniques* for a more accurate summary from a text. After that from a long meeting we'll generate summary and highlighting points.

# Software Components

List of Software/Hardware ToolsList of Software/Hardware Tools

Tool	Functions	Other similar Tools (if any)	Why selected this tool
Kaggle	Coding implementation	Jupyter Notebook, colab	lake of GPU in local device 2x 15 GB T4 GPU
Hugging Face	Model & Dataset Collection	Kaggle	Easy to select based on our requirements
Chat GPT	Compare summaries, bug fixing support	Stack Overflow	To debug & compare outputs
Transformers (Happy Transformer)	Library for Python		Open-source and offers a complete set of tools and pre-trained models for dealing with a variety of NLP applications
<i>google/pegasus-cn_dailymail</i>	Model used to train and generate summary		
<i>EleutherAI/gpt-neo/2.7B</i>	Model used to Fine-tune and generate summary		Previously selected PEGASUS model didn't work
<i>Microsoft/phi-2</i>	Model Used to generate summary using zero shot and few shot		Previously selected model couldn't generate a proper summary of meeting script

# Software Implementation

## Happy Transformer with EleutherAI/gpt-neo-125M

*Load the Happy Transformer with EleutherAI/gpt-neo-125M:*

```
[3] from happytransformer import HappyGeneration

[4] happy_gen = HappyGeneration("GPT-NEO", "EleutherAI/gpt-neo-125M")

/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens)
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(

config.json: 100% ██████████ 1.01k/1.01k [00:00<00:00, 59.6kB/s]
model.safetensors: 100% ██████████ 526M/526M [00:03<00:00, 162MB/s]
generation_config.json: 100% ██████████ 119/119 [00:00<00:00, 4.22kB/s]
tokenizer_config.json: 100% ██████████ 727/727 [00:00<00:00, 33.1kB/s]
vocab.json: 100% ██████████ 899k/899k [00:00<00:00, 3.58MB/s]
merges.txt: 100% ██████████ 456k/456k [00:00<00:00, 2.25MB/s]
tokenizer.json: 100% ██████████ 2.11M/2.11M [00:00<00:00, 11.8MB/s]
special_tokens_map.json: 100% ██████████ 357/357 [00:00<00:00, 16.4kB/s]
```

*Train for 25000 cell dataset:*

```
happy_gen.train('/kaggle/input/train25000/train25000.csv')
```

Generating train split: ██████████ 24999/0 [00:03<00:00, 8022.04 examples/s]

Map: 100% ██████████ 22499/22499 [00:49<00:00, 458.67 examples/s]

Map: 100% ██████████ 2500/2500 [00:05<00:00, 460.14 examples/s]

/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/\_functions.py:68: UserWarning:
The argument 'gather\_dim' is deprecated. Please use 'gather\_dims' instead.
 warnings.warn('Was asked to gather along dimension 0, but all ')

[11250/11250 4:27:28, Epoch 1/1]

Step	Training Loss	Validation Loss
1	5.754000	6.869854
1125	7.178200	8.763229
2250	4.963600	4.712391
3375	4.587700	5.139930
4500	4.743800	4.967206
5625	5.726100	4.385664
6750	6.027700	20.808872
7875	6.656600	4.411877
9000	5.432300	6.063212
10125	6.019000	5.828940
11250	5.525100	5.393856

+ Code + Markdown

```
[1]: args2 = GENSettings(min_length=70, max_length=100)
      result = happy_gen.generate_text("Liverpool target Neto is also wanted")
```



### *Results:*

```
[24]: args2 = GENSettings(min_length=70, max_length=100)
      result = happy_gen.generate_text("Liverpool target Neto is also wanted by PSG and clubs in Spain as Brendan Rodgers faces stiff comp
      print(result.text)
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
```

the and the and the and and the and the of the and the of the to and the and the and the and the quality the and the and the and the to the where the a  
nd the and the quality these and the all quality the and the also's the and the all of the all of of

```
[25]: args2 = GENSettings(min_length=70, max_length=100)
      result = happy_gen.generate_text("This is the moment that a crew of firefighters struggled to haul a giant pig out of a garden swim")
      print(result.text)
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
```

the tweeting also whereall qualityall qualityall qualityall qualityall qualityall quality site also the also also qualityall quality site qualityall qualityall also also also also also also qualityall qualityall also also also qualityall quality to also quality the also also also also alsos and the also the qualityall also quality to quality

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
```

[illegible]

+ Code

+ Markdown


# Phi-2


## *Load the model via pipeline (Zero shot Technique):*

```
from transformers import pipeline


model_name = "microsoft/phi-2"

pipe = pipeline(
    "text-generation", #s
    model=model_name,
    device_map="auto",
    trust_remote_code=True,
)
```


config.json: 100%  863/863 [00:00<00:00, 68.8kB/s]


configuration\_phi.py: 100%  9.26k/9.26k [00:00<00:00, 840kB/s]


A new version of the following files was downloaded from <https://huggingface.co/microsoft/phi-2>:  
- configuration\_phi.py  
- Make sure to double-check they do not contain any added malicious code. To avoid downloading new versions of the code file, you can pin a revision.


modeling\_phi.py: 100%  62.7k/62.7k [00:00<00:00, 772kB/s]


A new version of the following files was downloaded from <https://huggingface.co/microsoft/phi-2>:  
- modeling\_phi.py  
- Make sure to double-check they do not contain any added malicious code. To avoid downloading new versions of the code file, you can pin a revision.


modelsafetensors.index.json: 100%  35.7k/35.7k [00:00<00:00, 3.24MB/s]


Downloading shards: 100%  2/2 [00:59<00:00, 24.87s/it]


model-00001-of-00002.safetensors: 100%  5.00G/5.00G [00:57<00:00, 386MB/s]


model-00002-of-00002.safetensors: 100%  564M/564M [00:01<00:00, 385MB/s]


Loading checkpoint shards: 100%  2/2 [00:09<00:00, 4.22s/it]


generation\_config.json: 100%  124/124 [00:00<00:00, 11.4kB/s]

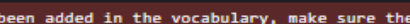
tokenizer\_config.json: 100%  7.34k/7.34k [00:00<00:00, 632kB/s]

vocab.json: 100%  798k/798k [00:00<00:00, 29.5MB/s]

merges.txt: 100%  456k/456k [00:00<00:00, 27.2MB/s]

tokenizer.json: 100%  2.11M/2.11M [00:00<00:00, 6.38MB/s]

added\_tokens.json: 100%  1.08k/1.08k [00:00<00:00, 90.3kB/s]

special\_tokens\_map.json: 100%  99.0/99.0 [00:00<00:00, 7.65kB/s]

Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.

[+ Code](#) [+ Markdown](#)

## *Tokenizer and casual language model:*


```
from transformers import AutoTokenizer, AutoModelForCausalLM
import torch
```

[+ Code](#) [+ Markdown](#)

```
tokenizer = AutoTokenizer.from_pretrained(
    "microsoft/phi-2",
    trust_remote_code = True
)
```

Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.

```
model = AutoModelForCausalLM.from_pretrained(
    "microsoft/phi-2",
    torch_dtype = "auto",
    device_map = "auto",
    trust_remote_code = True
)
```

Loading checkpoint shards: 100%  2/2 [00:02<00:00, 1.17s/it]

## Output Code:

```
prompt = "Give me a short summary of the following text. Asana and Avir are twins who j
with torch.no_grad():
    token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
    output_ids = model.generate(
        token_ids.to(model.device),
        max_new_tokens=512,
        do_sample=True,
        temperature = 0.3
    )

output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
print(output)
```

## Testing

```
[15]: prompt = "Give me a short summary of the following text. One hot day, a thirsty crow flew all over the fields looking f
with torch.no_grad():
    token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
    output_ids = model.generate(
        token_ids.to(model.device),
        max_new_tokens=512,
        do_sample=True,
        temperature = 0.3
    )
    output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
    print(output)
```

The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's `attention\_mask` to obtain reliable results.

Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

OUTPUT: A thirsty crow found a water jug but couldn't drink from it due to its narrow neck. He used pebbles to raise the water level and quenched his thirst.

</endoftext>

```
[16]: prompt = "Give me a short summary of the following text. Hare was much amused at the idea of running a race with the To
with torch.no_grad():
    token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
    output_ids = model.generate(
        token_ids.to(model.device),
        max_new_tokens=512,
        do_sample=True,
        temperature = 0.3
    )

    output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
    print(output)
```

The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's `attention\_mask` to obtain reliable results.

Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

ANSWER: The text is a fable about a race between a hare and a tortoise. The hare is overconfident and lazy, while the tortoise is slow but steady. The hare wastes time by sleeping, while the tortoise keeps moving. The tortoise wins the race and teaches the hare a lesson.

</endoftext>

```
[18]: prompt = "Give me a short summary of the following text. Midas was a king of great fortune who ruled the country of Phrygia with torch.no_grad():
token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
output_ids = model.generate(
    token_ids.to(model.device),
    max_new_tokens=512,
    do_sample=True,
    temperature = 0.3
)

output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
print(output)
```

The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's `attention\_mask` to obtain reliable results.  
Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

A: Midas, a wealthy king, wished for everything he touched to turn into gold. Dionysus granted his wish, but Midas' daughter turned into a statue when he hugged her. Midas washed his hands in the river Pactolus and everything returned to normal. He became a kind and generous king, and his people mourned for him when he died.  
</endoftext>

```
[19]: prompt = "Give me a short summary of the following text. Portia had her first ever job interview. She had practiced answers with torch.no_grad():
token_ids = tokenizer.encode(prompt, add_special_tokens=False ,return_tensors="pt")
output_ids = model.generate(
    token_ids.to(model.device),
    max_new_tokens=1000,
    do_sample=True,
    temperature = 0.3
)

output = tokenizer.decode(output_ids[0][token_ids.size(1) :])
print(output)
```

The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass your input's `attention\_mask` to obtain reliable results.  
Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

ANSWER: Portia experienced a nervousness during her job interview, causing her to struggle to answer the questions. Her brother Joseph offered her a ride, but she had difficulty responding to his question.  
</endoftext>

## Comparison

Firstly, we tried to make a summary of a meeting script but the problem we faced was due to the massive length of the script which can not be taken as input by pegasus-cnn-dailymail model. To overcome this issue, we used a new model named Eleuthernet/gpt-neo-2.7B but faced issues with the resource. We used happy transformer library to load this model but couldn't load due to resource constraints. So, we tried to load the same model with 125 Million parameters and fine-tuned it with CNN Dataset and used the fine-tuned model to generate summary. But, the model didn't perform well on meeting transcript and didn't generate a summary. Then, we tried to use another model named "microsoft/phi-2" model. And, used the model with zero shot and few shot approach and generated summary. This model generates a proper summary of the meeting script.