```
from transformers import pipeline
summarizer = pipeline("summarization")
text="""Hugging face is a company that specializes in natural language processing(NLP).it has developed the Transformers libary,which pr
summary=summarizer(text,max_length=50,min_length=10,do_sample=False)
print(summary)
```

```
No model was supplied, defaulted to sshleifer/distilbart-cnn-12-6 and revision a4f8f3e (https://huggingface.co/sshleifer/distilbart
Using a pipeline without specifying a model name and revision in production is not recommended.
config.json: 100%                                     1.80k/1.80k [00:00<00:00, 78.5kB/s]
pytorch_model.bin: 100%                                   1.22G/1.22G [00:17<00:00, 70.5MB/s]
model.safetensors:  89%                                   1.09G/1.22G [00:14<00:01, 113MB/s]
tokenizer_config.json: 100%                                 26.0/26.0 [00:00<00:00, 627B/s]
vocab.json: 100%                                      899k/899k [00:00<00:00, 4.27MB/s]
merges.txt: 100%                                      456k/456k [00:00<00:00, 7.57MB/s]
Device set to use cpu
[{'summary_text': ' Hugging face is a company that specializes in natural language processing . It has developed the Transformers li
```

```
import torch
from transformers import AutoModelForSequenceClassification, AutoTokenizer
model_name = "cardiffnlp/tweet-topic-21-multi"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForSequenceClassification.from_pretrained(model_name)
labels = [
    "arts_&_culture", "business_&_entrepreneurs", "celebrity_&_pop_culture", "diaries_&_daily_life",
    "family", "fashion_&_style", "film_tv_&_video", "fitness_&_health", "food_&_dining",
    "gaming", "learning_&_educational", "music", "news_&_social_concern", "other_hobbies",
    "relationships", "science_&_technology", "sports_&_esports", "travel_&_adventure",
    "youth_&_student_life"
]
```

```
tokenizer_config.json:  100%                          1.30k/1.30k  [00:00<00:00,  14.8kB/s]
vocab.json:  100%                                     798k/798k  [00:00<00:00,  6.49MB/s]
merges.txt:  100%                                     456k/456k  [00:00<00:00,  8.83MB/s]
tokenizer.json:  100%                                 1.36M/1.36M  [00:00<00:00,  16.6MB/s]
special_tokens_map.json:  100%                        239/239  [00:00<00:00,  3.57kB/s]
config.json:  100%                                    1.88k/1.88k  [00:00<00:00,  36.5kB/s]
pytorch_model.bin:  100%                              499M/499M  [00:08<00:00,  19.3MB/s]
```

```python
import torch
from transformers import AutoTokenizer,AutoModelForSequenceClassification,AutoTokenizer

model_name="cardiffnlp/tweet-topic-21-multi"
tokenizer=AutoTokenizer.from_pretrained(model_name)
model=AutoModelForSequenceClassification.from_pretrained(model_name)

labels=[
    "arts_&culture","business&entrepreneurs","celebrity&pop_culture","diaries&_daily_life",
    "family","fashion_&style","film_tv&video","fitness&health","food&_dining",
    "gaming","learning_&educational","music","news&_social_concern","other_hobbies","relationships",
    "science_&technology","sports&esports","travel&adventure","youth&_student_life"
]

texts=[
    "The latest iPhone was just released with an incredible new camera!",
    "Manchester United won their match with a stunning goal in the last minute."
    "NASA just launched a new mission to explore the surface of Mars.",
    "The Oscars had some surprising winners this year!"
]

inputs=tokenizer(texts,padding=True,truncation=True,return_tensors="pt")
with torch.no_grad():
      outputs=model(**inputs)
```

```
probabilities=torch.nn.functional.softmax(outputs.logits,dim=-1)
predictions=torch.argmax(probabilities,dim=1)

for text,pred,prob in zip(texts,predictions,probabilities):
    print(f"Text: {text}\nTopic: {labels[pred.item()]}, Confidence: {prob[pred].item():.4f}\n")
```

```
Asking to truncate to max_length but no maximum length is provided and the model has no predefined maximum length. Default to no tru
Text: The latest iPhone was just released with an incredible new camera!
Topic: science_&technology, Confidence: 0.9260

Text: Manchester United won their match with a stunning goal in the last minute.NASA just launched a new mission to explore the surf
Topic: sports&esports, Confidence: 0.7513

Text: The Oscars had some surprising winners this year!
Topic: film_tv&video, Confidence: 0.9357
```

```
from transformers import AutoModelForCausalLM, AutoTokenizer
model_name = "gpt2"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)
prompt = "I love coding languages,"
inputs = tokenizer(prompt, return_tensors="pt")
output = model.generate(**inputs, max_length=50, num_return_sequences=1, temperature=0.7, top_k=50)
generated_text = tokenizer.decode(output[0], skip_special_tokens=True)
print(generated_text)
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
I love coding languages, but I'm not a programmer. I'm a programmer. I'm a programmer. I'm a programmer. I'm a programmer. I'm a pro
```

```
prompt="Hello, I'm alanguage model"
inputs=tokenizer(prompt,return_tensors="pt")
output=model.generate(**inputs,max_length=50,num_return_sequences=1,temperature=0.7,top_k=50)
```

```
generated_text=tokenizer.decode(output[0],skip_special_tokens=True)
print(generated_text)
```

```
/usr/local/lib/python3.11/dist-packages/transformers/generation/configuration_utils.py:628: UserWarning: `do_sample` is set to `Fals
  warnings.warn(
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Hello, I'm alanguage modeler. I'm a programmer. I'm a programmer. I'm a programmer. I'm a programmer. I'm a programmer. I'm a progra
```

```
import os
import atexit
import shutil
from transformers import BlenderbotTokenizer, BlenderbotForConditionalGeneration
model_name = "facebook/blenderbot-1B-distill"
tokenizer = BlenderbotTokenizer.from_pretrained(model_name)
model = BlenderbotForConditionalGeneration.from_pretrained(model_name)
def interact_with_chatbot(user_input, conversation_history):
    conversation_history.append(f"User: {user_input}")
    inputs = tokenizer(conversation_history, return_tensors="pt", padding=True, truncation=True)
    outputs = model.generate(**inputs, max_length=100, num_return_sequences=1)

    outputs = model.generate(**inputs, max_length=100, num_return_sequences=1)
```

```
tokenizer_config.json:  100%                    1.05k/1.05k  [00:00<00:00,  23.8kB/s]
```

```
Start coding or generate with AI.
```

```
merges.txt:  100%                          62.9k/62.9k  [00:00<00:00,  1.47MB/s]

added_tokens.json:  100%                        16.0/16.0  [00:00<00:00,  399B/s]

special_tokens_map.json:  100%                    772/772  [00:00<00:00,  25.0kB/s]

tokenizer.json:  100%                          310k/310k  [00:00<00:00,  4.18MB/s]

config.json:  100%                           1.38k/1.38k  [00:00<00:00,  37.0kB/s]

pytorch_model.bin:  100%                       2.87G/2.87G  [00:53<00:00,  60.4MB/s]

model.safetensors:   40%                       1.15G/2.87G  [00:20<00:30,  56.9MB/s]
```