L1: NLP tasks with a simple interface

Load your HF API key and relevant Python libraries.

In [*]:
```
import os
import io
from IPython.display import Image, display, HTML
from PIL import Image
import base64
from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file
hf_api_key = os.environ['HF_API_KEY']
```

In [2]:
```
# Helper function
import requests, json

#Summarization endpoint
def get_completion(inputs, parameters=None,ENDPOINT_URL=os.environ['HF_API_SUMMARY_BA
    headers = {
      "Authorization": f"Bearer {hf_api_key}",
      "Content-Type": "application/json"
    }
    data = { "inputs": inputs }
    if parameters is not None:
        data.update({"parameters": parameters})
    response = requests.request("POST",
                                ENDPOINT_URL, headers=headers,
                                data=json.dumps(data)
                               )
    return json.loads(response.content.decode("utf-8"))
```

Building a text summarization app

Here we are using an Inference Endpoint (https://huggingface.co/inference-endpoints) for the `shleifer/distilbart-cnn-12-6` , a 306M parameter distilled model from `facebook/bart-large-cnn` .

How about running it locally?

The code would look very similar if you were running it locally instead of from an API. The same is true for all the models in the rest of the course, make sure to check the Pipelines (https://huggingface.co/docs/transformers/main_classes/pipelines) documentation page

```
from transformers import pipeline

get_completion = pipeline("summarization", model="shleifer/distilbart-cnn-12-6")

def summarize(input):
    output = get_completion(input)
```

In [3]:
```
text = ('''The tower is 324 metres (1,063 ft) tall, about the same height
        as an 81-storey building, and the tallest structure in Paris.
        Its base is square, measuring 125 metres (410 ft) on each side.
        During its construction, the Eiffel Tower surpassed the Washington
        Monument to become the tallest man-made structure in the world,
        a title it held for 41 years until the Chrysler Building
        in New York City was finished in 1930. It was the first structure
        to reach a height of 300 metres. Due to the addition of a broadcasting
        aerial at the top of the tower in 1957, it is now taller than the
        Chrysler Building by 5.2 metres (17 ft). Excluding transmitters, the
        Eiffel Tower is the second tallest free-standing structure in France
        after the Millau Viaduct.''')

get_completion(text)
```

```
[{'summary_text': ' The tower is 324 metres (1,063 ft) tall, about the same height as
an 81-storey building . It is the tallest structure in Paris and the second tallest fr
ee-standing structure in France after the Millau Viaduct . It was the first structure
in the world to reach a height of 300 metres .'}]
```

Getting started with Gradio `gr.Interface`

How about running it locally?

The code would look very similar if you were running it locally. Simply remove all the paramters in the launch method

```
demo.launch()
```

In [4]:
```
import gradio as gr
def summarize(input):
    output = get_completion(input)
    return output[0]['summary_text']

gr.close_all()
demo = gr.Interface(fn=summarize, inputs="text", outputs="text")
demo.launch(share=True, server_port=int(os.environ['PORT1']))
```

```
Running on local URL:  https://0.0.0.0:39648 (https://0.0.0.0:39648)

Could not create share link. Missing file: /usr/local/lib/python3.9/site-packages/grad
io/frpc_linux_amd64_v0.2.

Please check your internet connection. This can happen if your antivirus software bloc
ks the download of this file. You can install manually by following these steps:

1. Download this file: https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd
64 (https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd64)
2. Rename the downloaded file to: frpc_linux_amd64_v0.2
3. Move the file to this location: /usr/local/lib/python3.9/site-packages/gradio
```

502 Bad Gateway

---

openresty

You can add `demo.launch(share=True)` to create a public link to share with your team or friends.

In [5]:
```
import gradio as gr

def summarize(input):
    output = get_completion(input)
    return output[0]['summary_text']

gr.close_all()
demo = gr.Interface(fn=summarize,
                    inputs=[gr.Textbox(label="Text to summarize", lines=6)],
                    outputs=[gr.Textbox(label="Result", lines=3)],
                    title="Text summarization with distilbart-cnn",
                    description="Summarize any text using the `shleifer/distilbart-cn
                   )
demo.launch(share=True, server_port=int(os.environ['PORT2']))
```

```
Closing server running on port: 39648
Running on local URL:  https://0.0.0.0:47506 (https://0.0.0.0:47506)

Could not create share link. Missing file: /usr/local/lib/python3.9/site-packages/grad
io/frpc_linux_amd64_v0.2.

Please check your internet connection. This can happen if your antivirus software bloc
ks the download of this file. You can install manually by following these steps:

1. Download this file: https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd
64 (https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd64)
2. Rename the downloaded file to: frpc_linux_amd64_v0.2
3. Move the file to this location: /usr/local/lib/python3.9/site-packages/gradio
```

502 Bad Gateway

openresty

Building a Named Entity Recognition app

We are using this Inference Endpoint (https://huggingface.co/inference-endpoints) for `dslim/bert-base-NER` , a 108M parameter fine-tuned BART model on the NER task.

How about running it locally?

```
from transformers import pipeline

get_completion = pipeline("ner", model="dslim/bert-base-NER")

def ner(input):
    output = get_completion(input)
    return {"text": input, "entities": output}
```

In [6]:
```
API_URL = os.environ['HF_API_NER_BASE'] #NER endpoint
text = "My name is Andrew, I'm building DeepLearningAI and I live in California"
get_completion(text, parameters=None, ENDPOINT_URL= API_URL)
```


```
[{'entity': 'B-PER',
  'score': 0.9990625,
  'index': 4,
  'word': 'Andrew',
  'start': 11,
  'end': 17},
 {'entity': 'B-ORG',
  'score': 0.9927857,
  'index': 10,
  'word': 'Deep',
  'start': 32,
  'end': 36},
 {'entity': 'I-ORG',
  'score': 0.99677867,
  'index': 11,
  'word': '##L',
  'start': 36,
  'end': 37},
 {'entity': 'I-ORG',
  'score': 0.9954496,
  'index': 12,
  'word': '##ear',
  'start': 37,
  'end': 40},
 {'entity': 'I-ORG',
  'score': 0.9959293,
  'index': 13,
  'word': '##ning',
  'start': 40,
  'end': 44},
 {'entity': 'I-ORG',
  'score': 0.8917463,
  'index': 14,
  'word': '##A',
  'start': 44,
  'end': 45},
 {'entity': 'I-ORG',
  'score': 0.50361204,
  'index': 15,
  'word': '##I',
  'start': 45,
  'end': 46},
 {'entity': 'B-LOC',
  'score': 0.99969244,
  'index': 20,
  'word': 'California',
  'start': 61,
  'end': 71}]
```

In [7]:
```
def ner(input):
    output = get_completion(input, parameters=None, ENDPOINT_URL=API_URL)
    return {"text": input, "entities": output}

gr.close_all()
demo = gr.Interface(fn=ner,
                    inputs=[gr.Textbox(label="Text to find entities", lines=2)],
                    outputs=[gr.HighlightedText(label="Text with entities")],
                    title="NER with dslim/bert-base-NER",
                    description="Find entities using the `dslim/bert-base-NER` model
                    allow_flagging="never",
                    #Here we introduce a new tag, examples, easy to use examples for
                    examples=["My name is Andrew and I live in California", "My name
demo.launch(share=True, server_port=int(os.environ['PORT3']))
```

```
Closing server running on port: 39648
Closing server running on port: 47506
Running on local URL:  https://0.0.0.0:41617 (https://0.0.0.0:41617)

Could not create share link. Missing file: /usr/local/lib/python3.9/site-packages/grad
io/frpc_linux_amd64_v0.2.

Please check your internet connection. This can happen if your antivirus software bloc
ks the download of this file. You can install manually by following these steps:

1. Download this file: https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd
64 (https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd64)
2. Rename the downloaded file to: frpc_linux_amd64_v0.2
3. Move the file to this location: /usr/local/lib/python3.9/site-packages/gradio
```

502 Bad Gateway

---

openresty

Adding a helper function to merge tokens

In [*]:
```
def merge_tokens(tokens):
    merged_tokens = []
    for token in tokens:
        if merged_tokens and token['entity'].startswith('I-') and merged_tokens[-1]['
            # If current token continues the entity of the last one, merge them
            last_token = merged_tokens[-1]
            last_token['word'] += token['word'].replace('##', '')
            last_token['end'] = token['end']
            last_token['score'] = (last_token['score'] + token['score']) / 2
        else:
            # Otherwise, add the token to the list
            merged_tokens.append(token)

    return merged_tokens

def ner(input):
    output = get_completion(input, parameters=None, ENDPOINT_URL=API_URL)
    merged_tokens = merge_tokens(output)
    return {"text": input, "entities": merged_tokens}

gr.close_all()
demo = gr.Interface(fn=ner,
                    inputs=[gr.Textbox(label="Text to find entities", lines=2)],
                    outputs=[gr.HighlightedText(label="Text with entities")],
                    title="NER with dslim/bert-base-NER",
                    description="Find entities using the `dslim/bert-base-NER` model
                    allow_flagging="never",
                    examples=["My name is Andrew, I'm building DeeplearningAI and I l

demo.launch(share=True, server_port=int(os.environ['PORT4']))
```

```
Closing server running on port: 39648
Closing server running on port: 47506
Closing server running on port: 41617
Running on local URL:  https://0.0.0.0:57171 (https://0.0.0.0:57171)
```

In [*]:
```
gr.close_all()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: