Professional Training Program in Large Language Models

Day 4

Hamza Farooq
Founder & CEO
Traversaal.ai

SDAIA
الهيئة السعودية للبيانات والذكاء الاصطناعي
Saudi Data & AI Authority

Day 4: Introduction to Hugging Face

Learning outcomes

- What is Hugging Face
- Why is it so important for the AI Community
- Analyzing the Notebook
  - Encoder Models
  - Inference APIs
  - Agentic Frameworks
  - Pipelines

Firstly, what is Hugging Face?

Hugging Face (https://huggingface.co/) is a company dedicated to the building, deployment and application of LLM Tools and the publishing of LLM Models.

Hugging Face: an LLM hangout spot

Anyone with an email address can register in HuggingFace and take advantage of its myriad of services, APIs, demonstration boxes and take a look at the best models for each category (encoding, text generation, image generat…)

Hugging Face: an LLM hangout spot

Hugging Face is very well known in the AI community for the sharing of different kinds of models.

- Did you build your own LLM?
- Did you finetune it on your own training data?
- Do you wanna see how it performs compared to the others?

Hugging Face: an LLM hangout spot

Post your model and you will have it graded against all the other submitted models, here

https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard

T	Model	Average	IFEval	BBH	MATH Lvl 5	GPQA	MUSR	MMLU-
💬	MaziyarPanahi/calme-2.1-qwen2-72b	43.61	81.63	57.33	36.03	17.45	20.15	49.05
💬	Qwen/Qwen2-72B-Instruct	42.49	79.89	57.48	35.12	16.33	17.17	48.92
🔶	pankajmathur/orca_mini_v7_72b	39.06	59.3	55.06	26.44	18.01	24.21	51.35
💬	MaziyarPanahi/calme-2.2-llama3-70b	37.98	82.08	48.57	22.96	12.19	15.3	46.74
🔶	VAGOsolutions/Llama-3-SauerkrautLM-70b-Instruct	37.82	80.45	52.03	21.68	10.4	13.54	48.8
🔶	ValiantLabs/Llama3-70B-Fireplace	36.82	77.74	49.56	19.64	13.98	16.77	43.25
💬	meta-llama/Meta-Llama-3-70B-Instruct	36.18	80.99	50.19	23.34	4.92	10.92	46.74
💬	meta-llama/Meta-Llama-3.1-70B-Instruct	35.97	84.28	54.45	2.72	8.84	17.32	48.19
🔶	failspy/llama-3-70B-Instruct-abliterated	35.79	80.23	48.94	23.72	5.26	10.53	46.06
🔶	Sao10K/L3-70B-Euryale-v2.1	35.35	73.84	48.7	20.85	10.85	12.25	45.6
🔶	migtissera/Llama-3-70B-Synthia-v3.5	35.2	60.76	49.12	18.96	18.34	23.39	40.65

Hugging Face: an LLM hangout spot

Models are ranked based on a variety of statistics; the values you see on the right are basic benchmarks that score how good the model does on various tasks (mathematical tasks, basic reasoning, context awareness…)

Hugging Face: an LLM hangout spot

Most of these models are Open-Source; they can be downloaded and used locally, without the need of payment or an API key.

Even excellent models like Mixtral 8x7b are downloadable freely… just make sure to have a big enough machine!

Hugging Face: an LLM hangout spot

Each model is provided with its own page. The model’s publishers describe it in as much detail as possible, and most of the times you can try it out on a small text-area on the right side!

MaziyarPanahi / **calme-2.1-qwen2-72b** ♡ like 15

Text Generation | Transformers | Safetensors | English | qwen2 | chat | qwen | finetune | chatml | conversational | Eval Results | text-generation-inference

License: tongyi-qianwen (other)

Model card | Files and versions | Community 16 | Train | Deploy | Use this model

Edit model card

MaziyarPanahi/calme-2.1-qwen2-72b

This model is a fine-tuned version of the powerful `Qwen/Qwen2-72B-Instruct`

Downloads last month
865

Safetensors | Model size 72.7B params | Tensor type BF16

Text Generation

Inference API (serverless) has been turned off for this model.

Finetuned from `Qwen/Qwen2-72B-Instruct`

Collections including `MaziyarPanahi/calme-2.1-qwen2-72b`

Fine-Tuned Collection

40 items • Updated 3 days ago • △ 6

Hugging Face: The Transformer Library

In the programming community, HuggingFace is best known for one of the most famous AI repositories out there: ***transformers***.

https://github.com/huggingface/transformers

Hugging Face: The Transformer Library

With the transformers library, you can download **AND TRAIN** state-of-the-art pretrained models, reducing compute costs and saving time and resources, instead of training a model from scratch

Hugging Face: The Transformer Library

NOTE: Beware, the Transformers library is updated very VERY frequently, so code that you have written a month ago may NOT work due to package updates

huggingface / transformers Public

Notifications Fork 25.8k Star 130k

Code Issues 948 Pull requests 265 Actions Projects 26 Security Insights

main 497 Branches 169 Tags Go to file Code

zucchini-nlp Flash-Attn: fix generation when no attention mask or no pa... 81233c0 · 2 hours ago 16,454 Commits

.circleci	pytest_num_workers=4 for some CircleCI jobs (#31764)	3 weeks ago
.github	Bug report update -- round 2 (#32006)	last week
benchmark	Benchmark GitHub Actions workflow (#31163)	last month
docker	Docker: TF pin on the consistency job (#31928)	2 weeks ago
docs	Fix code snippet for Grounding DINO (#32229)	17 hours ago
examples	Refactor: Removed un-necessary object base class (#32...	3 hours ago
i18n	docs: move translations to i18n (#31584)	last month
model_cards	Update URL for Hub PR docs (#17532)	2 years ago
notebooks	[Docs] Add missing language options and fix broken links	5 months ago

About

🤗 Transformers: State-of-the-art Machine Learning for Pytorch, TensorFlow, and JAX.

huggingface.co/transformers

python nlp machine-learning natural-language-processing deep-learning tensorflow pytorch transformer speech-recognition seq2seq flax pretrained-models language-models nlp-library language-model hacktoberfest bert jax pytorch-transformers model-hub

Readme

Apache-2.0 license

ON TO OUR NOTEBOOK

Hugging Face: The Notebook

The Notebook is divided into three different sections:

- Basic Sentence operation one can do with HuggingFace
- An example of how you can leverage Agents for Image Generation
- Generating text with a model pipeline

Hugging Face Notebook: Section 1

Firstly, navigate to this site and make an account:

https://huggingface.co/join

Once you have done that, navigate to the Settings section, and make an API key

Hugging Face Notebook: Section 1

Why the API key?

You will need it to download different models, and use specific tools.

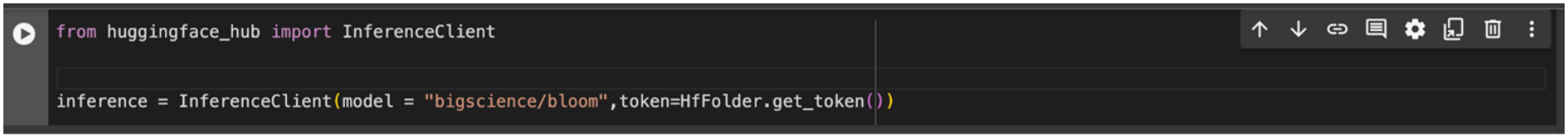
Hugging Face Notebook: Section 1

Covered in this section:

- **Sentence Completion**
- **Question Answering** → Based on Context
- **Summarization** → Based on Context

Hugging Face Notebook: Sentence Completion

Sentence Completion refers to the act of completing an initiated sentence. It is a simple task, and many models can do it.



This line calls upon a HuggingFace API (InferenceClient). The API will use the specific model (bigscience/bloom) remotely; the model will not be hosted on your machine, but on the HuggingFace servers.

Hugging Face Notebook: Sentence Completion

For each model, you can initialize many different parameters to change the WAY your model will answer questions/complete questions/perform different tasks

```
top_k = None if top_k == 0 else top_k
do_sample = False if num_beams > 0 else not greedy_decoding
num_beams = None if (greedy_decoding or num_beams == 0) else num_beams
no_repeat_ngram_size = None if num_beams is None else no_repeat_ngram_size
top_p = None if num_beams else top_p
early_stopping = None if num_beams is None else num_beams > 0

params = {
    "max_new_tokens": max_length,
    "top_k": top_k,
    "top_p": top_p,
    "temperature": temperature,
    "do_sample": do_sample,
    "seed": seed,
    "early_stopping":early_stopping,
    "no_repeat_ngram_size":no_repeat_ngram_size,
    "num_beams":num_beams,
    "return_full_text":return_full_text
}
```

Hugging Face Notebook: Sentence Completion

Why did it stop before finishing a real sentence?

Because we asked it to deliver a maximum of *max_length* tokens, which maps to the number of characters outputted.

Note: The number of tokens is NOT the number of characters, but more tokens does mean more characters

```
prompt = "The thing that makes large language models interesting is"
resp = json.loads(infer(prompt))

resp[0]['generated_text']
```

```
'The thing that makes large language models interesting is that they are very large. The largest models are in the hundreds of billions of paramet
ers. This is'
```

Hugging Face Notebook: Question Answering

You can use some models to answer questions based on some context you have provided for them?

Hugging Face Notebook: Question Answering

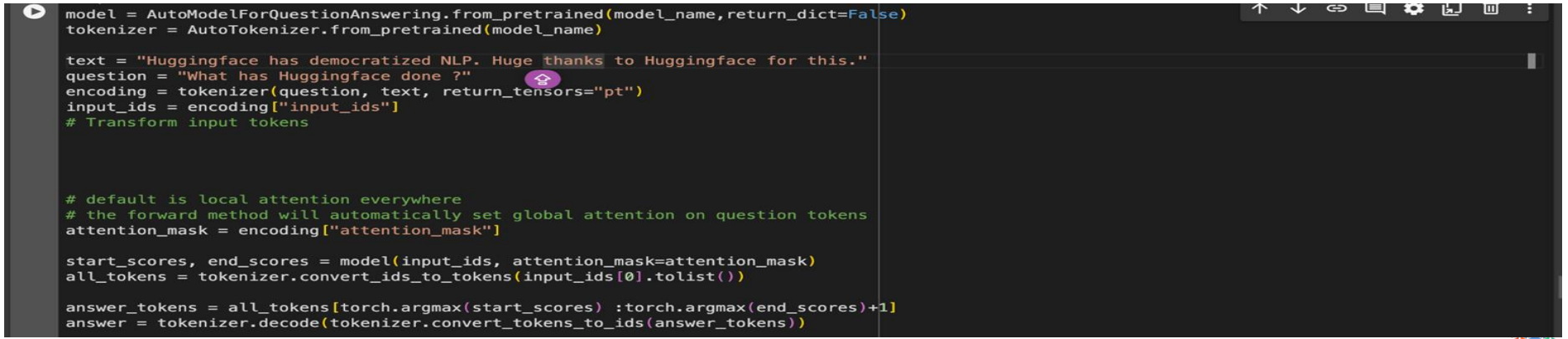
What does this mean?

It means you must have two sets of things:

1. Your question, and an embedding model designed FOR QUESTIONS
2. Your context, from which the model will derive the answer from, and an embedding model designed FOR CONTEXT

Hugging Face Notebook: Question Answering

Note: HuggingFace offers functions that combine the two embedding models together, like in this case, where you only need one call instead of 2



```
model = AutoModelForQuestionAnswering.from_pretrained(model_name,return_dict=False)
tokenizer = AutoTokenizer.from_pretrained(model_name)

text = "Huggingface has democratized NLP. Huge thanks to Huggingface for this."
question = "What has Huggingface done ?"
encoding = tokenizer(question, text, return_tensors="pt")
input_ids = encoding["input_ids"]
# Transform input tokens



# default is local attention everywhere
# the forward method will automatically set global attention on question tokens
attention_mask = encoding["attention_mask"]

start_scores, end_scores = model(input_ids, attention_mask=attention_mask)
all_tokens = tokenizer.convert_ids_to_tokens(input_ids[0].tolist())

answer_tokens = all_tokens[torch.argmax(start_scores) :torch.argmax(end_scores)+1]
answer = tokenizer.decode(tokenizer.convert_tokens_to_ids(answer_tokens))
```

Hugging Face Notebook: Question Answering

And the answer

```
[ ] answer
```

```
' democratized NLP'
```

Hugging Face Notebook: Summarization

Summarization, **with context**, refers to summarizing a corpus of text.

Hugging Face Notebook: Summarization

This is very similar to the Question answering with context, and follows the same concept, however it does not need a question (since the task is summarizing).

Hugging Face Notebook: Summarization

It does need, however, a summarization model from HuggingFace

Hugging Face Notebook: Summarization

It does need, however, a summarization model from HuggingFace.

In this case, the API is used through a web request.

```
API_URL = "https://api-inference.huggingface.co/models/facebook/bart-large-cnn"

def query(payload):
    response = requests.post(API_URL, json=payload)
    return response.json()

params = {'do_sample': False}

full_text = '''AI applications are summarizing articles, writing stories and
engaging in long conversations — and large language models are doing
the heavy lifting.

A large language model, or LLM, is a deep learning model that can
understand, learn, summarize, translate, predict, and generate text and other
content based on knowledge gained from massive datasets.

Large language models – successful applications of
transformer models. They aren't just for teaching AIs human languages,
but for understanding proteins, writing software code, and much, much more.

In addition to accelerating natural language processing applications —
like translation, chatbots, and AI assistants — large language models are
used in healthcare, software development, and use cases in many other fields.'''

output = query({
    'inputs': full_text,
    'parameters': params
})
```

Hugging Face Notebook: Summarization

The output

```
[{'summary_text': 'A large language model, or LLM, is a deep learning model '
                  'that can understand, learn, summarize, translate, predict, '
                  'and generate text. They aren’t just for teaching AIs human '
                  'languages, but for understanding proteins, writing software '
                  'code, and much more.'}]
```

Hugging Face Notebook: Section 2

Before we dive into the code, let’s very briefly introduce agents

Hugging Face Notebook: Agents

You can think of an LLM Agent in a Framework in the following manner:

You are building a beautiful University. You have builders, sculptors, architects and painters. Each one has a hyper specific task they need to perform

Hugging Face Notebook: Agents

An Agent is an LLM assistant that performs VERY SPECIFIC actions.

It could be image generation, it could be sentence completion, or code generation (and execution of that code).

Hugging Face Notebook: Agents

In this case, we are using an agent to call upon an image generation based on our request.

Think of it like this:

You are a wealthy businessman and are tasking the best painter in your city to paint your new castle! The painter is the agent, while the brush is the image tool.

Hugging Face Notebook: Agents

Our painter, for this exercise, is Mixtral 8X7B, on a remote machine hosted by HuggingFace.

We are going to supply it with an image generation model called m-ric/text-to-image

Hugging Face Notebook: Agents

Mixtral will then generate python code to USE the image tool.

It will execute the code, and we will have our image!

Hugging Face Notebook: Agents

Utilizing the concept of agents, you can come up with a myriad of solutions for MANY tasks to solve!

Hugging Face Notebook: Section 3

The third section relates to the *pipeline* module.

Hugging Face Notebook: Pipeline

The pipeline performs the following operations, all in two line:

1. Model download
2. Parameter setting
3. Query tokenization
4. Query answering

Hugging Face Notebook: Pipeline

In other words, you can answer questions very simply. This is the closest ‘Chat GPT’-esque behaviour that is present in this code.

Hugging Face Notebook: Pipeline

In other words, you can answer questions very simply. This is the closest ‘Chat GPT’-esque behaviour that is present in this code.

Hugging Face Notebook: Pipeline

First line, local model download and parameter setting

```
import torch
from transformers import pipeline

generate_text = pipeline(model="databricks/dolly-v2-3b", torch_dtype=torch.bfloat16, trust_remote_code=True, device_map="auto")
```

Second line, question answering

```
res = generate_text("Explain to me the difference between nuclear fission and fusion.")
print(res[0]["generated_text"])
```

```
In nuclear fusion, atomic nuclei merge to form a very high atomic number, heavy element called a nuclide. Atoms from one nuclide are fused with nuclei from another,

There are several fusion approaches:
- magnetic confinement fusion where the fusing nuclei are held by a magnetic field
- tokamak fusion where the fusing nuclei are held inside of a toroidal magnetic field by a plasma
- coldfusion where a plasma held in a toroidal magnetic field is used to support the ionisation of to nuclear fuel and accelerating the ions to collide to form the f
```

Appendix

THANK YOU!


SDAIA
الهيئة السعودية للبيانات
والذكاء الاصطناعي
Saudi Data & AI Authority