Uso de modelos de linguagem natural com python

primeiramente no Ubuntu criar um ambiente isolado para executar o python sem usar o sistema principal.

criei uma pasta /nlp_project e dentro dela:

```
sudo apt install python3-pip
sudo apt install python3-venv

cd /boot/nlp_project/
$/nlp_project$ python3 -m venv env
source env/bin/activate
$/nlp_project$ pip install torch torchvision torchaudio transformers




python3 -m venv env
source env/bin/activate

python test_cuda.py
True
1
NVIDIA GeForce GTX 1660 Ti
```

indicativo de que o torch está ativado e utilizando o cuda na placa de vídeo

```
mport torch
print(torch.cuda.is_available())  # Should print True
print(torch.cuda.device_count())  # Should show the number of GPUs
available
print(torch.cuda.get_device_name(0))  # Should print "NVIDIA GeForce GTX
1660 Ti"
```

executado código com modelo mais simples:

```
from transformers import pipeline

# Load the pipeline
generator = pipeline("text-generation", model="gpt2")
```

```
# Generate text
result = generator("How Earth came to be", max_length=50,
num_return_sequences=1)
print(result)
```

resultado

```
python face_model.py


Hardware accelerator e.g. GPU is available in the environment, but no
`device` argument is passed to the `Pipeline` object. Model will be on CPU.
Truncation was not explicitly activated but `max_length` is provided a
specific value, please use `truncation=True` to explicitly truncate
examples to max length. Defaulting to 'longest_first' truncation strategy.
If you encode pairs of sequences (GLUE-style) with the tokenizer you can
select this strategy more precisely by providing a specific strategy to
`truncation`.
Setting `pad_token_id` to `eos_token_id`:None for open-end generation.
[{'generated_text': "The quick brown fox that likes coffee isn't exactly
your typical bad boy.\n\nBut as the year of 2014 approaches, those of us
who can work ourselves into a mental state of disrepair should make a
conscious effort to take a moment to think"}]
```

depois um modelo mais complexo:

```
from transformers import pipeline

# Load the pipeline
generator = generator = pipeline("text-generation", model="EleutherAI/gpt-
neo-1.3B", device=0)


# Generate text
result = generator(
    "How Earth came to be and we are all still here",
    max_length=300,
    num_return_sequences=2,
    temperature=0.7,  # Creativity level
    top_p=0.9         # Nucleus sampling
)
print(result)
```

Com o resultado:

```
Truncation was not explicitly activated but `max_length` is provided a
specific value, please use `truncation=True` to explicitly truncate
examples to max length. Defaulting to 'longest_first' truncation strategy.
If you encode pairs of sequences (GLUE-style) with the tokenizer you can
select this strategy more precisely by providing a specific strategy to
`truncation`.
Setting `pad_token_id` to `eos_token_id`:None for open-end generation.
[{'generated_text': 'How Earth came to be the way it is today\n\nThe Big
Bang Theory is not a documentary. It is not an academic study of physics.
It is not a scientific paper. It is not an encyclopedia of the history of
science.\n\nIt is a comedy, an entertainment, an episode of the show that
we all know and love. And that is what it is.\n\nBut it is not just the big
bang theory that is the show. It is not just the big bang theory that is
the show.\n\nIt is a show that is about the science of the universe and the
origins of the universe. It is about how we understand the universe and the
universe is about how we understand the science of the universe.\n\nThe
show is about how the universe came to be the way it is today.\n\nThe show
is about the big bang theory and how the big bang theory came to be.\n\nAnd
it is about the big bang theory.\n\n'}, {'generated_text': 'How Earth came
to be\n\nThe evolution of Earth is a long and complex story. It began with
the Earth’s formation, about 4.6 billion years ago, and continues today
with the evolution of the planet’s crust and mantle, and the evolution of
life on Earth.\n\nIn the process of this evolution, the Earth has changed
and changed again. It has been a warm, wet, and often hot planet throughout
its history. It has had oceans and seas, and continents and continents. It
has had land, and sea, and continents. It has had lakes, and rivers, and
lakes. It has had rivers and lakes and forests and rainforests. It has had
continents and oceans. It has had continents and continents and oceans and
seas.\n\nThe Earth is a planet that has had many different kinds of life,
and many different kinds of life-forms. It has been a very warm, wet, and
often hot, wet planet throughout its'}]
```

Por fim um script com mais dados e log, resposta menor para não ficar muito incoerente e saber se está usando a placa de vídeo

```python
from transformers import pipeline, AutoTokenizer, AutoModelForCausalLM
import torch
import time

# Function to log GPU memory usage
def log_gpu_memory():
    if torch.cuda.is_available():
        print("\n[GPU MEMORY USAGE]")
        print(torch.cuda.memory_summary(device=0, abbreviated=False))
    else:
        print("\n[INFO] GPU not available. Using CPU.")
```

```
# Log the start time
start_time = time.time()

# Log: Initializing the pipeline
print("[INFO] Initializing the model and pipeline...")

# Load the model and tokenizer
model_name = "EleutherAI/gpt-neo-1.3B"
model = AutoModelForCausalLM.from_pretrained(model_name)
tokenizer = AutoTokenizer.from_pretrained(model_name)
generator = pipeline("text-generation", model=model, tokenizer=tokenizer,
device=0)

# Log: Loaded model details
log_gpu_memory()

# Tokenization check
input_text = "How Earth came to be and we are all still here"
tokens = tokenizer(input_text, return_tensors="pt")
print(f"\n[DEBUG] Input text: {input_text}")
print(f"[DEBUG] Tokenized input: {tokens}")

# Generate text
print("[INFO] Generating text...")
try:
    result = generator(
        input_text,
        max_length=100,
        num_return_sequences=2,
        temperature=0.7,  # Creativity level
        top_p=0.9         # Nucleus sampling
    )
    log_gpu_memory()
except RuntimeError as e:
    print("[ERROR] A RuntimeError occurred during generation.")
    print(str(e))
    torch.cuda.empty_cache()
    print("[INFO] Cleared GPU memory. Try reducing `max_length` or batch
size.")

# Print results
print("\n[RESULTS]")
for i, res in enumerate(result):
    print(f"Sequence {i + 1}:\n{res['generated_text']}\n")

# Log total time taken
end_time = time.time()
print(f"[INFO] Total time elapsed: {end_time - start_time:.2f} seconds")
```

com o resultado

```
python nlm_earth_model_eleutherAI_advance.py
[INFO] Initializing the model and pipeline...

[GPU MEMORY USAGE]
|===========================================================================
=|
|                  PyTorch CUDA memory summary, device ID 0
|
|---------------------------------------------------------------------------
-|
|            CUDA OOMs: 0            |        cudaMalloc retries: 0
|
|===========================================================================
=|
|        Metric         | Cur Usage  | Peak Usage | Tot Alloc  | Tot Freed
|
|---------------------------------------------------------------------------
-|
| Allocated memory      |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Active memory         |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Requested memory      |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| GPU reserved memory   |   5132 MiB |   5132 MiB |   5132 MiB |      0 B
|
|       from large pool |   5130 MiB |   5130 MiB |   5130 MiB |      0 B
|
|       from small pool |      2 MiB |      2 MiB |      2 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Non-releasable memory |  17884 KiB |  19808 KiB | 396656 KiB | 378772 KiB
|
```

```
|       from large pool |  17784 KiB |  17784 KiB | 394616 KiB | 376832 KiB
|
|       from small pool |    100 KiB |   2040 KiB |   2040 KiB |   1940 KiB
|
|---------------------------------------------------------------------------
-|
| Allocations           |     364    |     364    |     364    |       0
|
|       from large pool |     170    |     170    |     170    |       0
|
|       from small pool |     194    |     194    |     194    |       0
|
|---------------------------------------------------------------------------
-|
| Active allocs         |     364    |     364    |     364    |       0
|
|       from large pool |     170    |     170    |     170    |       0
|
|       from small pool |     194    |     194    |     194    |       0
|
|---------------------------------------------------------------------------
-|
| GPU reserved segments |     148    |     148    |     148    |       0
|
|       from large pool |     147    |     147    |     147    |       0
|
|       from small pool |       1    |       1    |       1    |       0
|
|---------------------------------------------------------------------------
-|
| Non-releasable allocs |       3    |       3    |      26    |      23
|
|       from large pool |       2    |       2    |      25    |      23
|
|       from small pool |       1    |       1    |       1    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize allocations  |       0    |       0    |       0    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize GPU segments |       0    |       0    |       0    |       0
|
|===========================================================================
=|


[DEBUG] Input text: How Earth came to be and we are all still here
[DEBUG] Tokenized input: {'input_ids': tensor([[2437, 3668, 1625,  284,
307,  290,  356,  389,  477,  991,  994]]), 'attention_mask': tensor([[1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])}
[INFO] Generating text...
Truncation was not explicitly activated but `max_length` is provided a
```

```
specific value, please use `truncation=True` to explicitly truncate
examples to max length. Defaulting to 'longest_first' truncation strategy.
If you encode pairs of sequences (GLUE-style) with the tokenizer you can
select this strategy more precisely by providing a specific strategy to
`truncation`.
Setting `pad_token_id` to `eos_token_id`:None for open-end generation.

[GPU MEMORY USAGE]
|===========================================================================
=|
|                  PyTorch CUDA memory summary, device ID 0
|
|---------------------------------------------------------------------------
-|
|            CUDA OOMs: 0            |        cudaMalloc retries: 0
|
|===========================================================================
=|
|        Metric         | Cur Usage  | Peak Usage | Tot Alloc  | Tot Freed
|
|---------------------------------------------------------------------------
-|
| Allocated memory      |   5122 MiB |   5205 MiB |  17949 MiB |  12826 MiB
|
|       from large pool |   5120 MiB |   5198 MiB |   7616 MiB |   2496 MiB
|
|       from small pool |      1 MiB |     54 MiB |  10332 MiB |  10330 MiB
|
|---------------------------------------------------------------------------
-|
| Active memory         |   5122 MiB |   5205 MiB |  17949 MiB |  12826 MiB
|
|       from large pool |   5120 MiB |   5198 MiB |   7616 MiB |   2496 MiB
|
|       from small pool |      1 MiB |     54 MiB |  10332 MiB |  10330 MiB
|
|---------------------------------------------------------------------------
-|
| Requested memory      |   5122 MiB |   5203 MiB |  17711 MiB |  12588 MiB
|
|       from large pool |   5120 MiB |   5196 MiB |   7387 MiB |   2267 MiB
|
|       from small pool |      1 MiB |     54 MiB |  10323 MiB |  10321 MiB
|
|---------------------------------------------------------------------------
-|
| GPU reserved memory   |   5270 MiB |   5270 MiB |   5270 MiB |      0 B
|
|       from large pool |   5210 MiB |   5210 MiB |   5210 MiB |      0 B
|
|       from small pool |     60 MiB |     60 MiB |     60 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
```

```
| Non-releasable memory |   9564 KiB |  72936 KiB |  13708 MiB |  13698 MiB
|
|       from large pool |   9464 KiB |  70792 KiB |   3033 MiB |   3023 MiB
|
|       from small pool |    100 KiB |  20480 KiB |  10674 MiB |  10674 MiB
|
|---------------------------------------------------------------------------
-|
| Allocations           |     365    |     444    |   77948    |   77583
|
|       from large pool |     171    |     220    |    1942    |    1771
|
|       from small pool |     194    |     273    |   76006    |   75812
|
|---------------------------------------------------------------------------
-|
| Active allocs         |     365    |     444    |   77948    |   77583
|
|       from large pool |     171    |     220    |    1942    |    1771
|
|       from small pool |     194    |     273    |   76006    |   75812
|
|---------------------------------------------------------------------------
-|
| GPU reserved segments |     181    |     181    |     181    |       0
|
|       from large pool |     151    |     151    |     151    |       0
|
|       from small pool |      30    |      30    |      30    |       0
|
|---------------------------------------------------------------------------
-|
| Non-releasable allocs |       3    |      42    |   34059    |   34056
|
|       from large pool |       2    |      23    |     518    |     516
|
|       from small pool |       1    |      40    |   33541    |   33540
|
|---------------------------------------------------------------------------
-|
| Oversize allocations  |       0    |       0    |       0    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize GPU segments |       0    |       0    |       0    |       0
|
|===========================================================================
=|


[RESULTS]
Sequence 1:
How Earth came to be and we are all still here
```

```
We are all still here. We are all still here.

This is the message that has been sent to us from our great and ancient
ancestors in the form of the message of the Great Mother, the Great Mother
of all that is, the Great Mother of our ancestors, the Great Mother of our
ancestors.

This is the message that has been sent to us from our great and ancient
ancestors in the form of the message

Sequence 2:
How Earth came to be and we are all still here

The story of our planet is a fascinating one. But what happens when we
begin to unravel the secrets of the origin of life on our planet?

This week, I was invited to give a talk on the topic of Earth’s origins at
the annual meeting of the American Geophysical Union (AGU) in San
Francisco.

The talk was part of a series of talks on the origin of life presented by
ge

[INFO] Total time elapsed: 4.93 seconds
```

Mudei o prompt para input_text = "Are forests in danger from global warming? And are they responsible for it?"

e o resultado:

```
python nlm_earth_model_eleutherAI_advance.py
[INFO] Initializing the model and pipeline...

[GPU MEMORY USAGE]
|===========================================================================
=|
|                  PyTorch CUDA memory summary, device ID 0
|
|---------------------------------------------------------------------------
-|
|            CUDA OOMs: 0            |        cudaMalloc retries: 0
|
|===========================================================================
=|
|        Metric         | Cur Usage  | Peak Usage | Tot Alloc  | Tot Freed
|
|---------------------------------------------------------------------------
-|
| Allocated memory      |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
```

```
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Active memory         |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Requested memory      |   5114 MiB |   5114 MiB |   5114 MiB |      0 B
|
|       from large pool |   5112 MiB |   5112 MiB |   5112 MiB |      0 B
|
|       from small pool |      1 MiB |      1 MiB |      1 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| GPU reserved memory   |   5132 MiB |   5132 MiB |   5132 MiB |      0 B
|
|       from large pool |   5130 MiB |   5130 MiB |   5130 MiB |      0 B
|
|       from small pool |      2 MiB |      2 MiB |      2 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Non-releasable memory |  17884 KiB |  19808 KiB | 396656 KiB | 378772 KiB
|
|       from large pool |  17784 KiB |  17784 KiB | 394616 KiB | 376832 KiB
|
|       from small pool |    100 KiB |   2040 KiB |   2040 KiB |   1940 KiB
|
|---------------------------------------------------------------------------
-|
| Allocations           |     364    |     364    |     364    |       0
|
|       from large pool |     170    |     170    |     170    |       0
|
|       from small pool |     194    |     194    |     194    |       0
|
|---------------------------------------------------------------------------
-|
| Active allocs         |     364    |     364    |     364    |       0
|
|       from large pool |     170    |     170    |     170    |       0
|
|       from small pool |     194    |     194    |     194    |       0
|
|---------------------------------------------------------------------------
```

```
-|
| GPU reserved segments |     148    |     148    |     148    |       0
|
|       from large pool |     147    |     147    |     147    |       0
|
|       from small pool |       1    |       1    |       1    |       0
|
|---------------------------------------------------------------------------
-|
| Non-releasable allocs |       3    |       3    |      26    |      23
|
|       from large pool |       2    |       2    |      25    |      23
|
|       from small pool |       1    |       1    |       1    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize allocations  |       0    |       0    |       0    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize GPU segments |       0    |       0    |       0    |       0
|
|===========================================================================
=|


[DEBUG] Input text: Are forests in danger from global warming? And are they
responsible for it?
[DEBUG] Tokenized input: {'input_ids': tensor([[ 8491, 17039,   287,  3514,
422,  3298,  9917,    30,   843,   389,
           484,  4497,   329,   340,    30]]), 'attention_mask':
tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])}
[INFO] Generating text...
Truncation was not explicitly activated but `max_length` is provided a
specific value, please use `truncation=True` to explicitly truncate
examples to max length. Defaulting to 'longest_first' truncation strategy.
If you encode pairs of sequences (GLUE-style) with the tokenizer you can
select this strategy more precisely by providing a specific strategy to
`truncation`.
Setting `pad_token_id` to `eos_token_id`:None for open-end generation.

[GPU MEMORY USAGE]
|===========================================================================
=|
|                  PyTorch CUDA memory summary, device ID 0
|
|---------------------------------------------------------------------------
-|
|            CUDA OOMs: 0            |        cudaMalloc retries: 0
|
|===========================================================================
=|
|        Metric         | Cur Usage  | Peak Usage | Tot Alloc  | Tot Freed
```

```
|
|---------------------------------------------------------------------------
-|
| Allocated memory      |   5122 MiB |   5205 MiB |  17596 MiB |  12474 MiB
|
|       from large pool |   5120 MiB |   5198 MiB |   7614 MiB |   2493 MiB
|
|       from small pool |      1 MiB |     54 MiB |   9982 MiB |   9980 MiB
|
|---------------------------------------------------------------------------
-|
| Active memory         |   5122 MiB |   5205 MiB |  17596 MiB |  12474 MiB
|
|       from large pool |   5120 MiB |   5198 MiB |   7614 MiB |   2493 MiB
|
|       from small pool |      1 MiB |     54 MiB |   9982 MiB |   9980 MiB
|
|---------------------------------------------------------------------------
-|
| Requested memory      |   5122 MiB |   5203 MiB |  17359 MiB |  12237 MiB
|
|       from large pool |   5120 MiB |   5196 MiB |   7386 MiB |   2265 MiB
|
|       from small pool |      1 MiB |     54 MiB |   9973 MiB |   9971 MiB
|
|---------------------------------------------------------------------------
-|
| GPU reserved memory   |   5270 MiB |   5270 MiB |   5270 MiB |      0 B
|
|       from large pool |   5210 MiB |   5210 MiB |   5210 MiB |      0 B
|
|       from small pool |     60 MiB |     60 MiB |     60 MiB |      0 B
|
|---------------------------------------------------------------------------
-|
| Non-releasable memory |   9564 KiB |  72936 KiB |  13329 MiB |  13319 MiB
|
|       from large pool |   9464 KiB |  70792 KiB |   3027 MiB |   3018 MiB
|
|       from small pool |    100 KiB |  18544 KiB |  10301 MiB |  10301 MiB
|
|---------------------------------------------------------------------------
-|
| Allocations           |     365    |     444    |   74464    |   74099
|
|       from large pool |     171    |     220    |    1938    |    1767
|
|       from small pool |     194    |     273    |   72526    |   72332
|
|---------------------------------------------------------------------------
-|
| Active allocs         |     365    |     444    |   74464    |   74099
|
|       from large pool |     171    |     220    |    1938    |    1767
```

```
|
|       from small pool |     194    |     273    |   72526    |   72332
|
|---------------------------------------------------------------------------
-|
| GPU reserved segments |     181    |     181    |     181    |       0
|
|       from large pool |     151    |     151    |     151    |       0
|
|       from small pool |      30    |      30    |      30    |       0
|
|---------------------------------------------------------------------------
-|
| Non-releasable allocs |       3    |      44    |   31928    |   31925
|
|       from large pool |       2    |      23    |     514    |     512
|
|       from small pool |       1    |      42    |   31414    |   31413
|
|---------------------------------------------------------------------------
-|
| Oversize allocations  |       0    |       0    |       0    |       0
|
|---------------------------------------------------------------------------
-|
| Oversize GPU segments |       0    |       0    |       0    |       0
|
|===========================================================================
=|


[RESULTS]
Sequence 1:
Are forests in danger from global warming? And are they responsible for it?

Forests are the largest and most diverse biomes on earth. They are home to
more than one third of all species. But they are being destroyed by climate
change.

The UN’s Intergovernmental Panel on Climate Change (IPCC) published its
latest report in 2015, which warned that the planet is warming at an
average rate of 3.6°C per century. The IPCC’s

Sequence 2:
Are forests in danger from global warming? And are they responsible for it?

Forests cover about 90 percent of the earth’s surface. They are a vital
source of oxygen and other essential nutrients, and also provide habitats
for animals and plants.

But the forests are also a source of carbon dioxide, a greenhouse gas, and
they are a leading cause of global warming.

The forests are also a major source of carbon dioxide, a greenhouse gas,
```

```
and they are a

[INFO] Total time elapsed: 5.20 seconds
```