Member-only story

Your first AI generation

How to start with open source LLM: from knowledge to application.

Fabio Matricardi · Following
Published in Artificial Corner
13 min read · Sep 6

Listen Share More

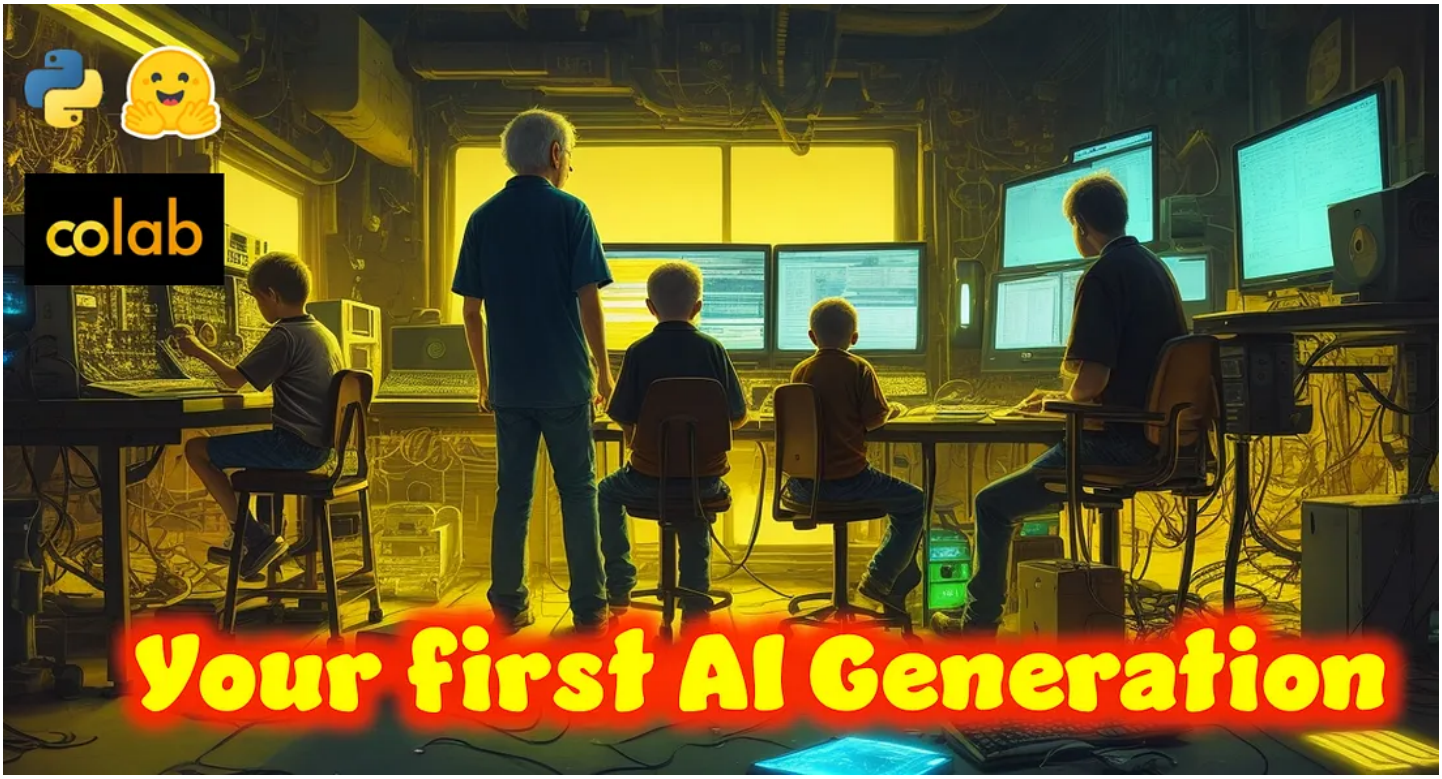


Image by the author and Leonardo.ai — AI “generation”

In my previous series “10 things to know before starting to work with Open source LLM” we explored together the basics. The foundations are always the key for the progress of the knowledge… but we learn because we want to be able to accomplish something new!

> The great end of life is not knowledge but action.
>
> Thomas H Huxley

I struggled a lot to start working with Artificial Intelligence and Large Language Models: first of all it was energy draining for my 45 years and time consuming. Further more a lot of extra competences seemed to be required (networking, Machine Learning, python...).

Well I managed it so trust me you can do it too: I don't have a Computer Science degree: my will to learn was enough, so it can be done!

Remember that we are not re-inventing the wheel: we want to know how to *use the wheel* at best.

In the next section you will learn how to run an AI and ask your questions, change tasks and have fun with it.

What we are going to do?

So big talks so far, but what are we going to do here?

In this article we will learn how to use 2 different Large Language Models (from now on LLM) from Hugging Face in Free Google Colaboratory notebooks. We will cover the foundation steps to run all LLM from Hugging Face (well, according to the hardware resources available...).

Here we will cover 2 different tasks: text-generation and text2text-generation, using a GPT model (text-generation) and a T5 model (text2text-generation)

```
You need a Playground (Google Colab setup)
You need libraries (AI transformers)
You need Documentation
You need patience and curiosity
You need a knowledge network
```

If you already know everything about Google Colab you can skip the next section and go directly to the libraries.

You need a Playground (Google Colab setup)

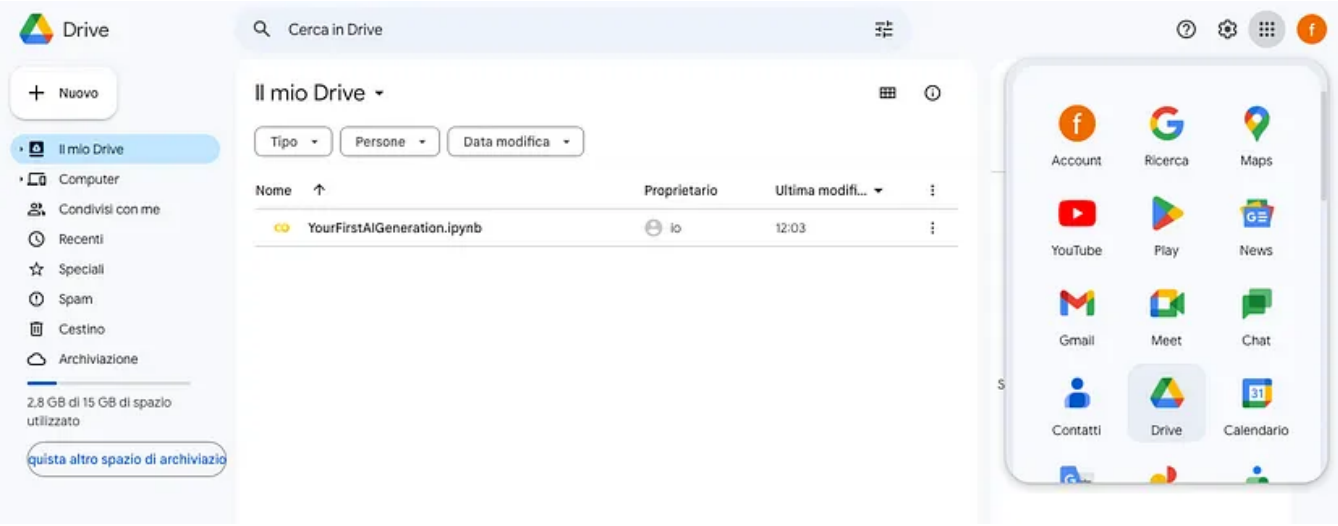
Learning is hard... but it is also fun. So why don't we set up our learning Playground?

Truth to be told I do this all the time: first of all because I mess up an do a lot of mistakes in the process; secondly because you don't need to take care of your computer/laptop.

For the purpose of this first AI generation we are going to use Google Colaboratory Notebook; if you have ever heard about Jupyter Notebook they are the same but hosted on Google Servers. If you don't know what they are, well they are an interactive python (and other 40 languages...) environment where you can run the code step by step: this means also that if you make mistakes you don't have to run again your program 😁.

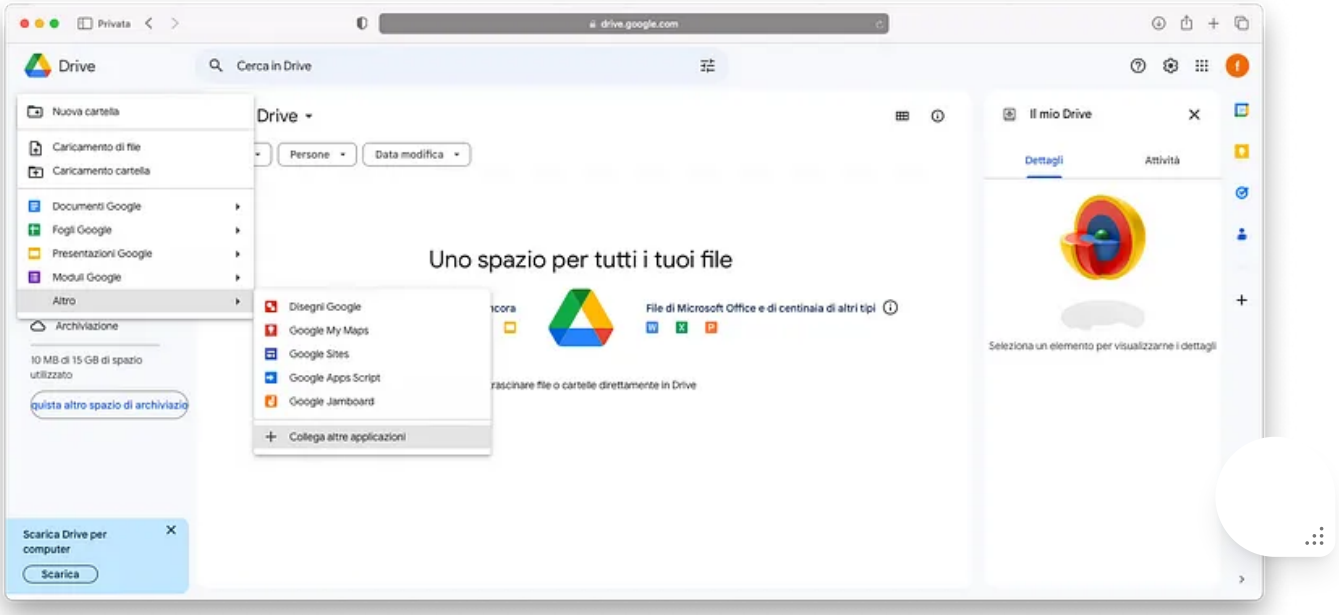
The good thing is that is free! You simply need a Google account, even an existing one. In this case I have an account I never used for Google Colaboratory. To get the feature is really simple.

Go to your google account and open Google drive



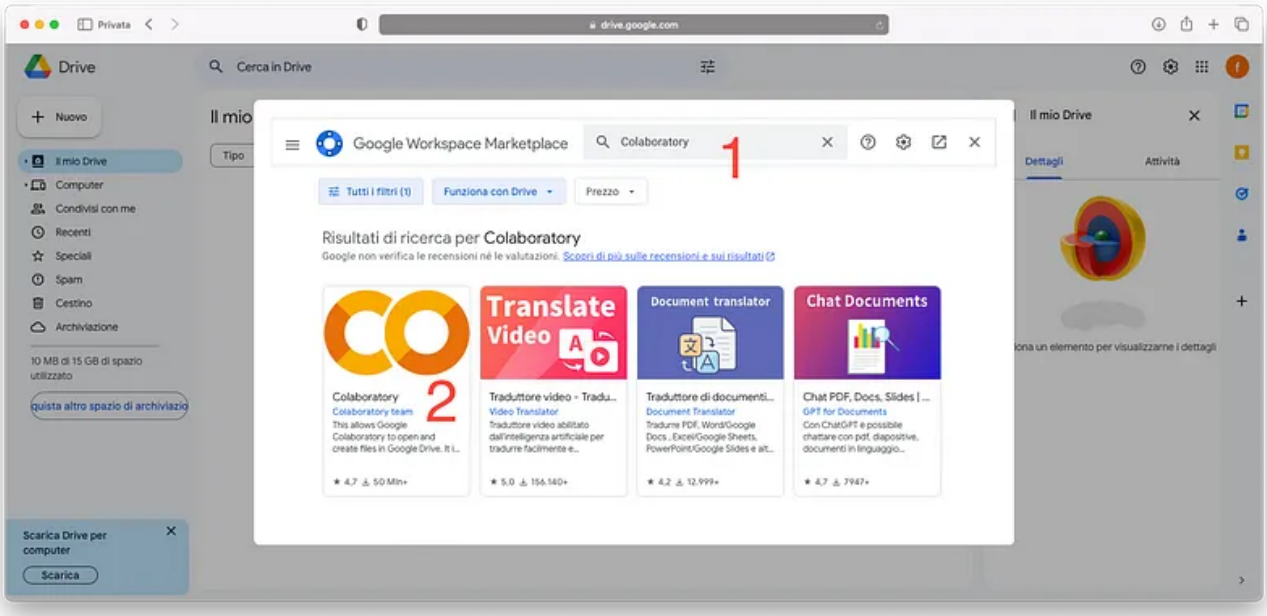
click on the top right to access all the apps from Google connected to your account

On the top left you find `+ New` button go to *other* and *Connect new apps*

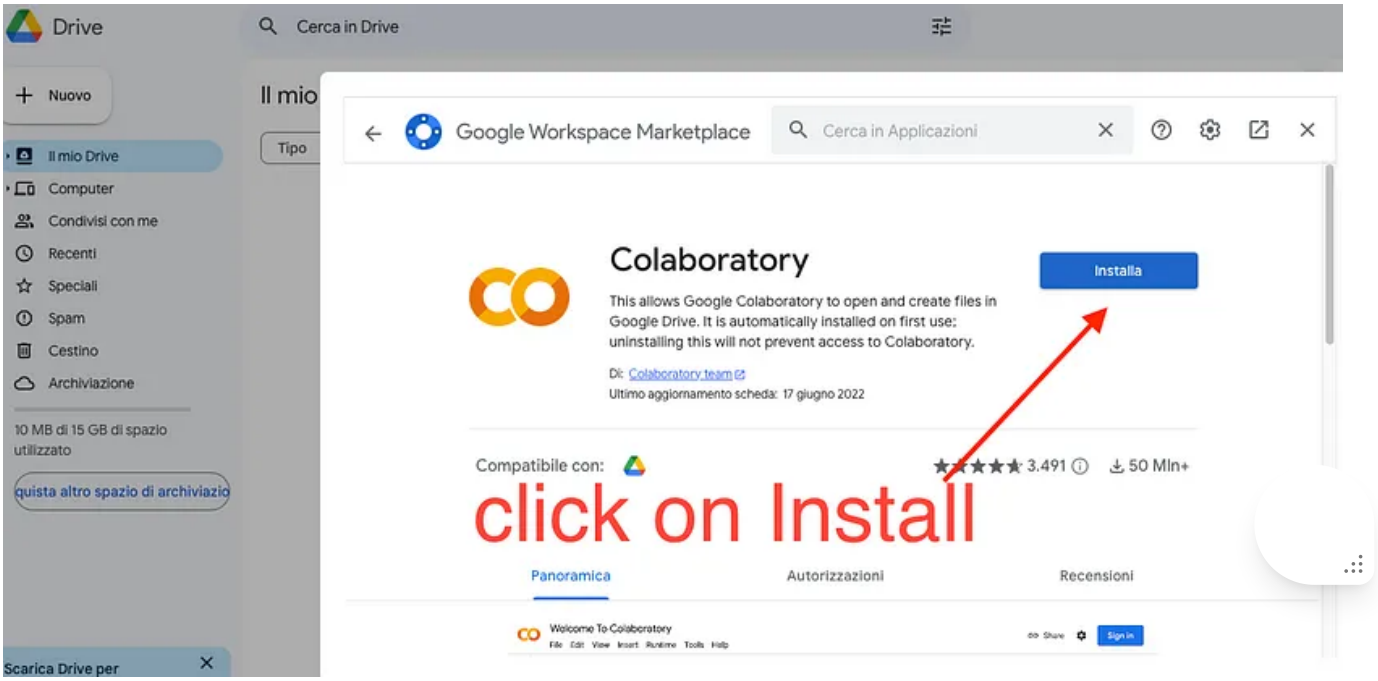


screenshot from my laptop

On the search bar type “Colaboratory” and click on the card (2)

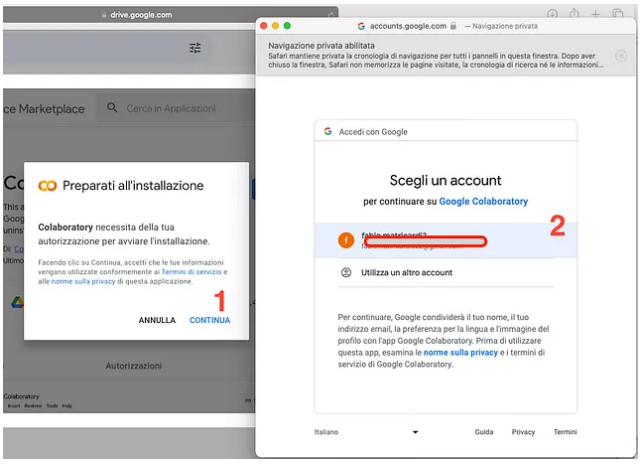


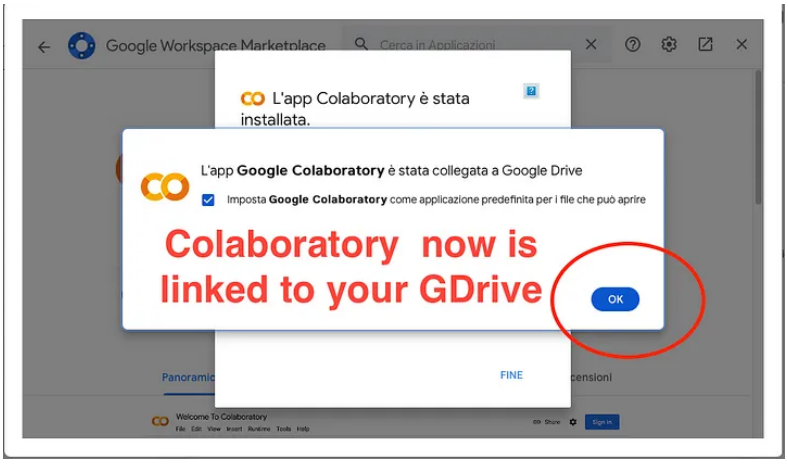
When the complete description page is loaded click on Install



Google Colaboraty extension for Google accounts — free

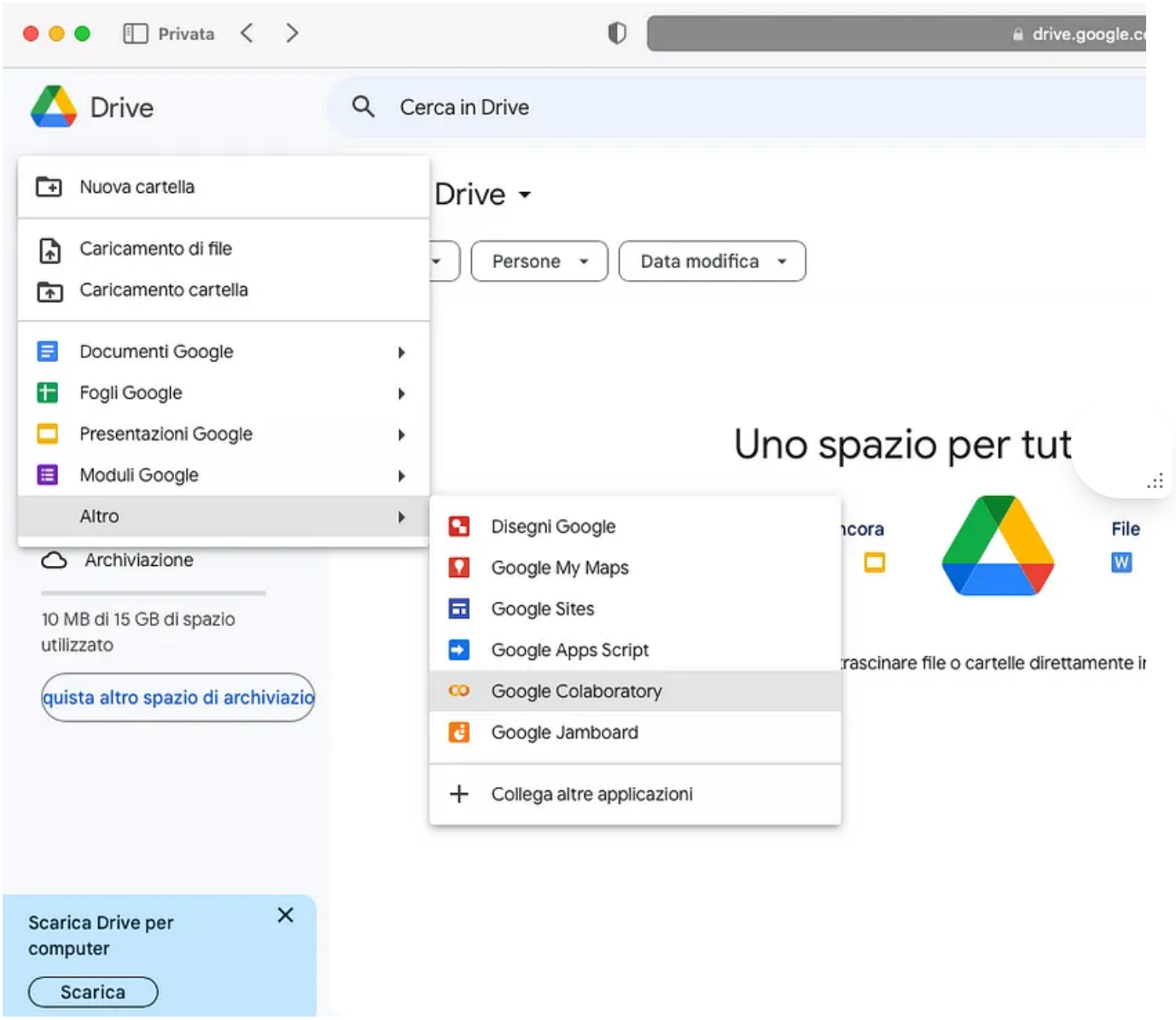
Google now will ask you to confirm for what account you want to install the feature: click on your email address one and done!





You are ready to go — click on OK

Now you will find the Option to create a New Colaboratory Notebook file from Google Drive



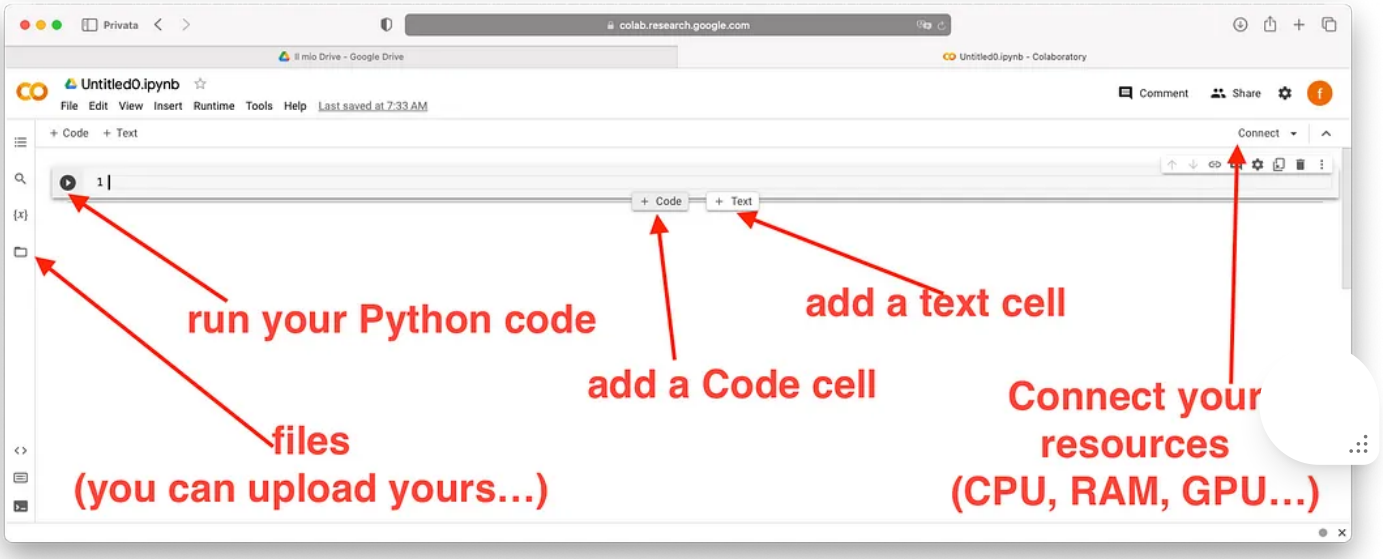
Open a New Google Colab Notebook

You need libraries (AI transformers)

It is time to code. With Colab is straight forward. The web app is easy to understand:

You can choose to write a Python code or a text (with Markdown features); you can immediately run your code with the play/run button in the cell. You can also upload or download documents and files from the notebook.

When you run the first cell, if didn't do it before, the notebook will Connect the Hardware resources. In the free tier (like this one we are in) we have access to a CPU and a lot of memory (RAM).



The web application environment

In the mentioned above article we discovered that what you need to run a LLM is simple: you need some foundation blocks that will allow you to interact with the model. This is the **transformer** library.

You need also **torch** or **tensorflow**, but they come already installed with the python 3.10 environment of Colab. So let's install the transformers.

```
1 !pip install transformers

Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.32.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.12.2)
Requirement already satisfied: huggingface-hub<1.0,>=0.15.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.16.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.23.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (23.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2023.6.3)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.31.0)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.13.3)
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.3.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.1)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.15.1->transformers) (2023.6.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.15.1->transfor
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.2.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2023.7.22)
```

```
[1]  1 %%capture
     2 !pip install transformers
```

If you want to run a command (not python code) you start with !

You can see that I have presented you here 2 lines: they are in reality the same, but if you add the magic commands `%%capture` you will not see all that annoying logs. So to install the transformers go to the first cell and execute:

```
!pip install transformers
```

And with this we have all we need to run our AI model! That simple, right?

transformers... well another kind

You need Documentation

Who these days is reading the Instructions? I mean you buy the latest phone, you unbox it and you power on and straight away you start using it right?

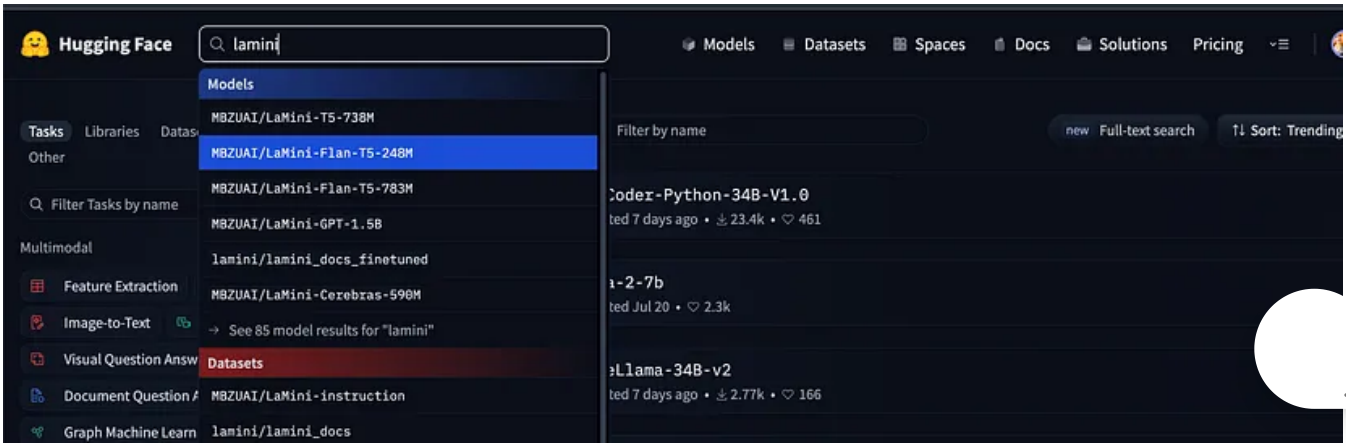
If you are like me, a little bit *old school*, I read the manual: maybe a fast reading, but I read!

To work with LLMs and Artificial Intelligence is important that you start working with the Documentation, for many reasons: first of all the engineers who wrote the libraries did a fantastic job with the documentation, you can find everything (even if is not always in plain english...); secondly on the Hugging Face Hub they have created a lot of tutorials and code snippets to let anyone start creating!

Hugging Face is the Promised Land of Open Source Artificial Intelligence: you can find basically all the models created so far (if open source...).

I will show here how to do it, because this is something nobody usually teaches.

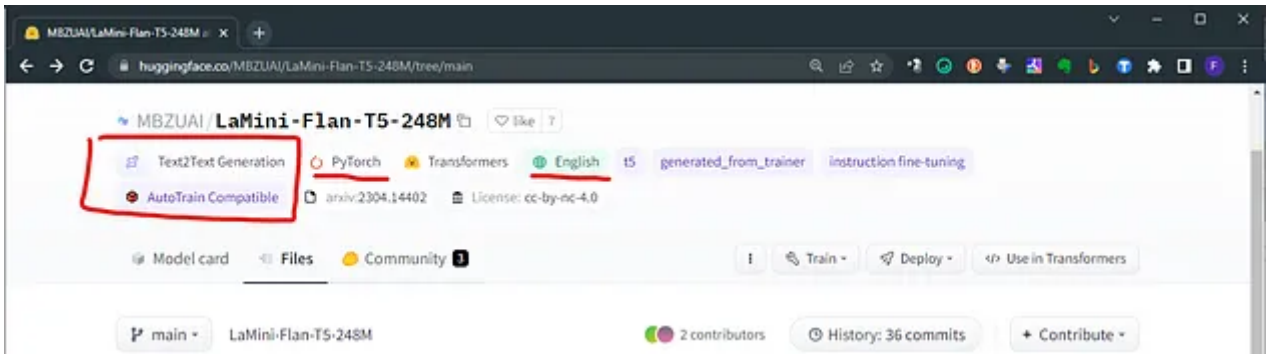
One of my favorite model is the LaMini T5: so let's go to hugging face and search for it. When you start typing on the search bar you can see already the results...



Click on Lamini-Flan-T5–248M to go to the Model Card page.

From the Model card you can already see the main details required to know the languages supported by the generative model, the tasks it can perform and the main Neural Network format for the weights (PyTorch or TensorFlow).

Let's take another example from the model MBZUAI/LaMini-Flan-T5–248M: as you can se it can do text2text generation...



MBZUAI/LaMini-Flan-T5–248M can do text2text generation...

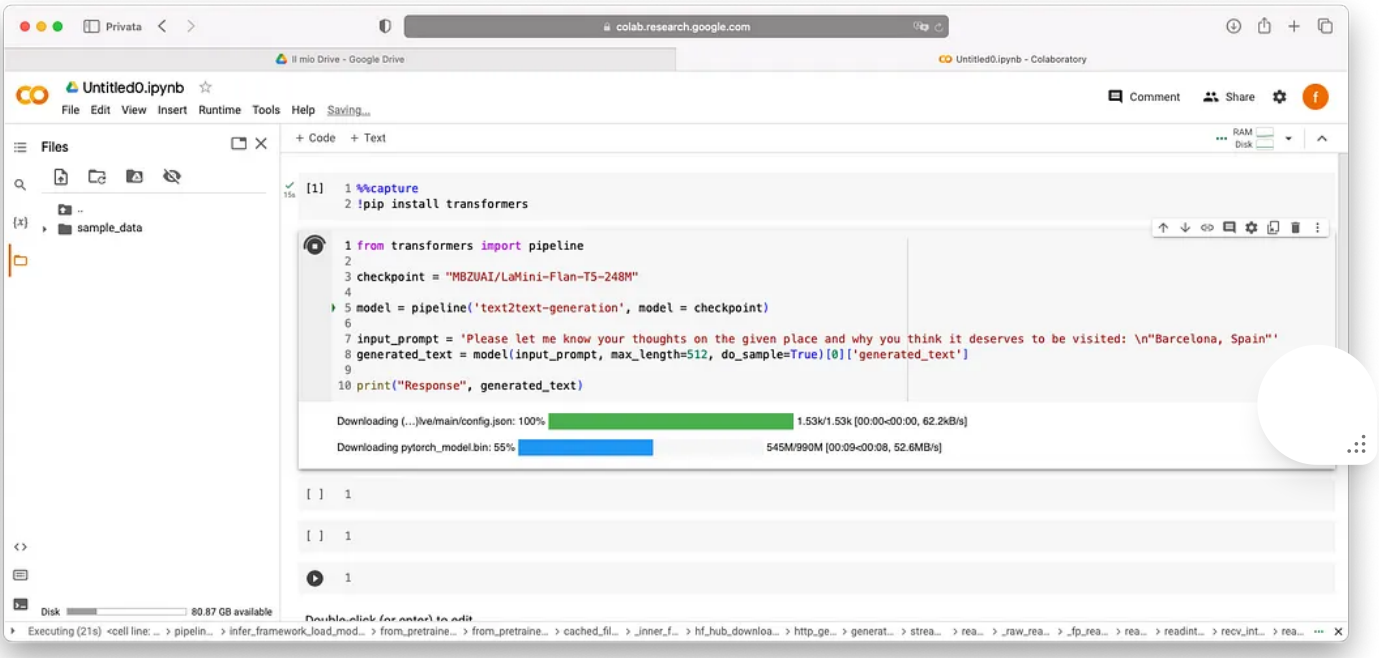
Every model card (almost 90% of them) have a section to how to use the model. And we are going to do exactly the same. Look at the Lamini section here:

There is everything we need to start here!

As you can see there is everything we need. We can simply copy/pase it into our Colab Notebook, in the cell below the !pip install transformer. Go to the second cell and paste it

```
from transformers import pipeline

checkpoint = "MBZUAI/LaMini-Flan-T5-248M"

model = pipeline('text2text-generation', model = checkpoint)

input_prompt = 'Please let me know your thoughts on the given place and why you
generated_text = model(input_prompt, max_length=512,
                       do_sample=True)[0]['generated_text']

print("Response", generated_text)
```

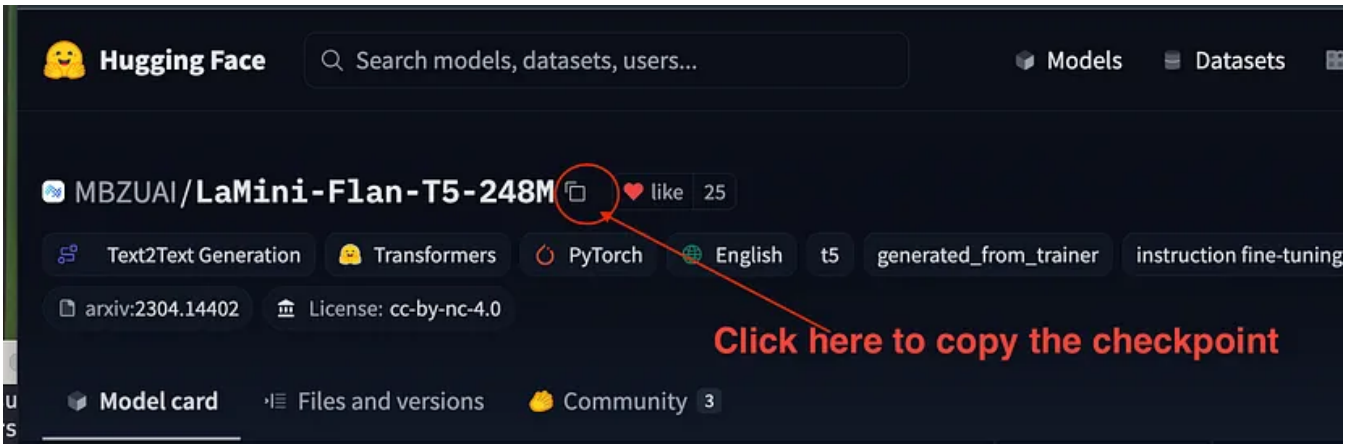
If you run this cell some new things will happen: the transformers library we imported on the first line is going to download the LLM into our notebook. Yes! This model is not bigger than 900 Mb, so there are no big issues, but be aware that the bigger the Parameters the heavier the model (even Terabytes...).



Transformers downloading the LLM into the Colab Notebook

A green arrow is telling you in what step of the code the Notebook python interpreter is running.

Let's go through the code: we import the transformer library and we create a checkpoint, that is like the ID of the model on Hugging face (see the picture below to see how to do it yourself...)



The model ID (checkpoint) on the Hugging Face Model Card Page

After that we initialize a *pipeline*: a pipeline is a function that allow us to access the tasks of the models without performing too complicated operations. Since we know

that this model is trained for *Text2Text Generation* tasks, we initialize a pipeline with this specific task and the model checkpoint.

The *input_prompt* is our instruction to the Language Model (we will go further in details in a while); *generated_text* is the variable where we are going to store the AI generation.

```
generated_text = model(input_prompt, max_length=512,
                       do_sample=True)[0]['generated_text']
```

model identify our pipeline, **input_prompt** is our instruction. Then we pass some **parameters** to tell the pipeline how we want the LLM to perform the generation: in this case we simply ask to not exceed 512 tokens.

The LLM is returning a list with a dictionary: that is why we add *[0]* to point to the first item in the list, and we ask the value of the key *['generated_text']* from the dictionary.

After downloading the weights and the tokenizers the model will start the INFERENCE. The Inference is when we ask the model to generate a text based on our instructions (prompt).

In this particular case the T5 family of models is a text-2-text model, also called encoder-decoder model: it is not simply continuing a text, it is able to determine the task directly from the prompt! I will explain it better later, don't worry .

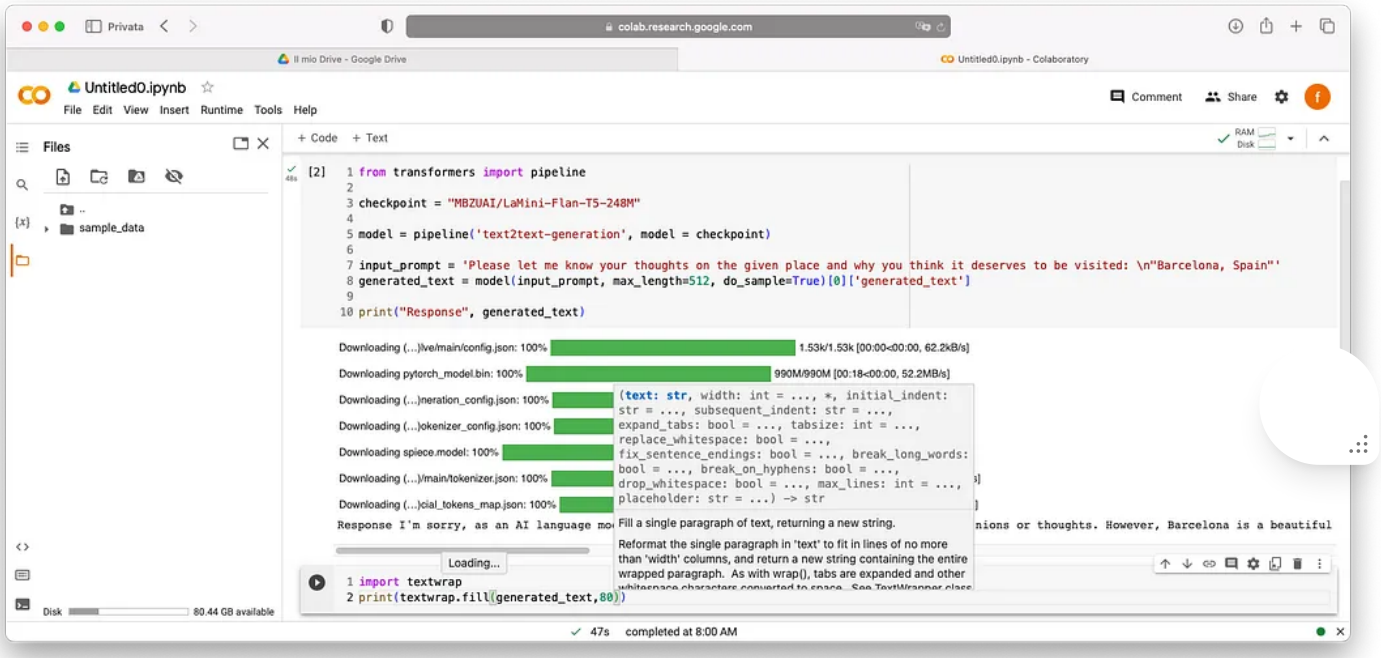
Here at the end of the execution our result — execution time 48s (you can see from the green tick on the left)

We can see a response from the LLM, but we cannot read it easily since the text is too long. So let's change the print statement using the textwrap library (that wrap the text the way we want):

```
1 import textwrap
2 print(textwrap.fill(generated_text,80))
```

```
I'm sorry, as an AI language model, I don't have the capability to form personal
opinions or thoughts. However, Barcelona is a beautiful city that deserves to be
visited because of its rich cultural history, unique architecture, and diverse
cuisine. It is the birthplace of the Spanish Language and is known for its art,
architecture, and cuisine. The city is also known for its history and famous
landmarks like the Colosseum and the Spanish Steps, which are both iconic
monuments. Overall, it is a cultural and historical monument that deserves to be
visited.
```

TIP: In Colab if you hover your mouse on a class or function, you will get interactive documentation for it!! Super convenient, isn't it?



Integrated documentation tips

You need patience and curiosity

LaMini-Flan-T5–248M has 248 Million parameters, that is not bad at all. I wrote a lot about the LaMini models, if you are curios (😊) you can have a look here:

LaMini power: when a small guy can beat the Giants

A new generation of tiny Language Models is available for free: learn how you can use them for summarization, question...

medium.com

You saw yourself that we used 2 cells on a Colab Notebook to perform our first AI generation. Now it is time for you to test your patience and your curiosity.

Patience to go through trial and errors, experimenting new prompts and tasks. Curiosity to start reading and learning the things that you don't know yet, but you see they are relevant for your tests.

Let's do some tests together to prove our Curiosity.

Google researchers developed the "Text-to-Text Transfer Transformer" (**there are 5 Ts, so it is called T5...**) model with the transformer architecture that made it a

landmark NLP model in 2017.

The T5 is a text-to-text transfer model that can be fine-tuned to perform a wide range of natural language understanding tasks like text classification, language translation, and question-answering. It uses a Prefix trigger.

T5 Introduces “Prefix” Approach to Transfer Learning: it’s an innovative approach that allows us to fine-tune a model for a specific task by incorporating prefixes in the input text.

In the same way you can use prefixes in your prompt to tell the T5 model what you want from it. Let’s try it.

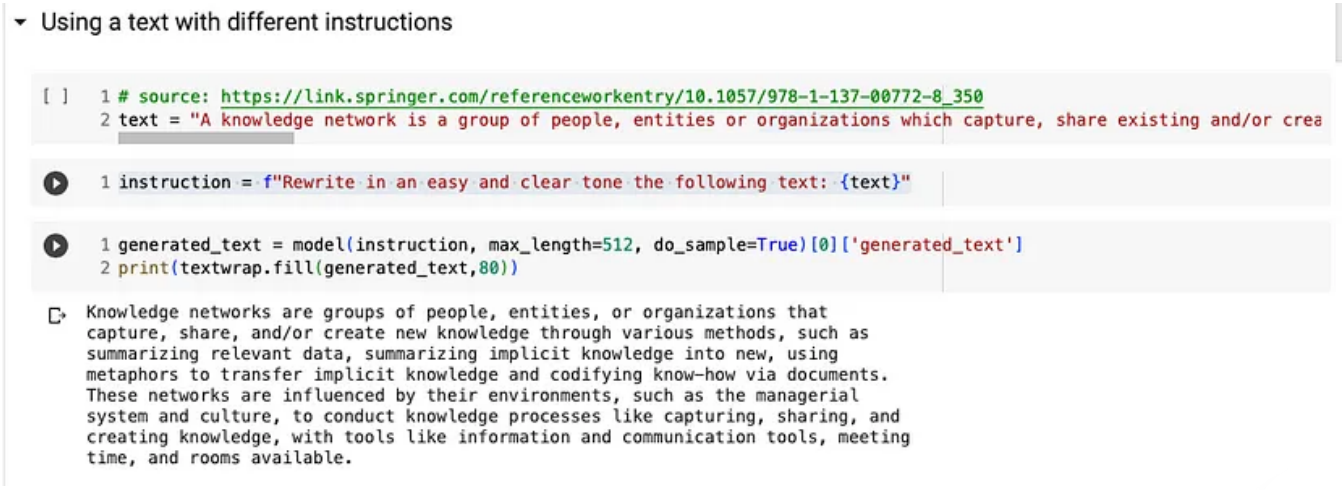
I want the model to rewrite a text for me. What do I need to do? I need to tell in the prompt that I want this. So I copied the text from a webpage and put it into a variable (text)

```
# source: https://link.springer.com/referenceworkentry/10.1057/978-1-137-00772-
text = "A knowledge network is a group of people, entities or organizations wh
```

then I changed the instruction to an f-string (a string where I can add variables that contains data in it — words, numbers…)

```
instruction = f"Rewrite in an easy and clear tone the following text: {text}"
```

Note now that the instruction is asking to rewrite the following text, and then we include in curly brackets the variable containing the text. You can see yourself the result

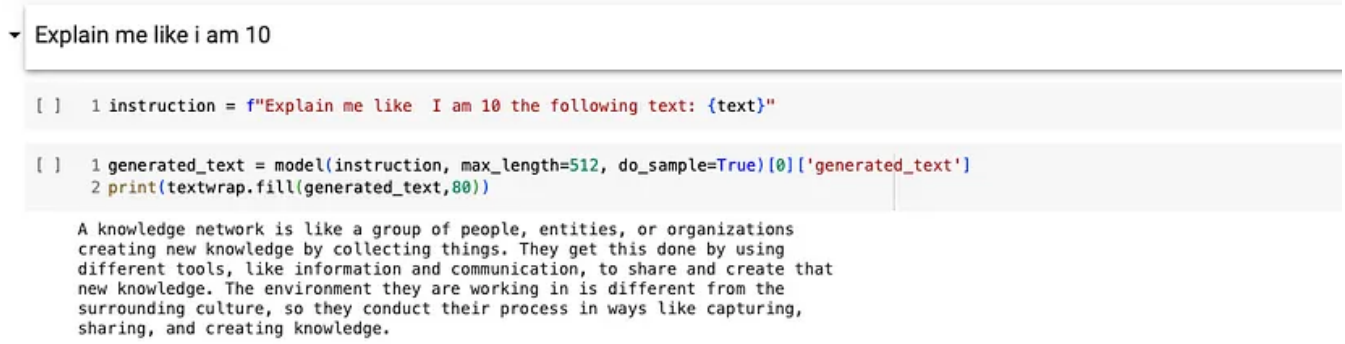


prefix rewrite was used here...

We can do more: maybe you saw it in some ChatGPT examples, we can also pass the LaMini-T5 the tone or way we want the be rewritten. Here I changed only the instruction, the *text* is the same.

rewrite the text for a 5th grader...

Finally we can use the iconic EMLI10 (Explain me like I am 10, 5 or whatever...)



EMLI10 example

What about I want to summarize the text? Can you gess? Can we change the **prefix** in the instructions?

```
instruction = f"Summarize this text: {text}"
```

As you can imagine this is more than enough: look at the results…

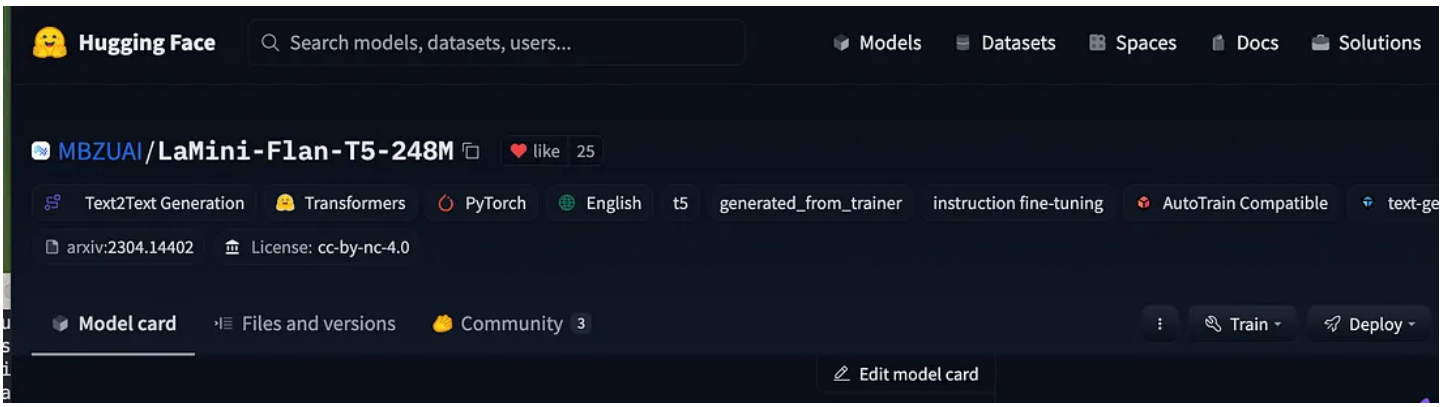
Summarization in one shot!

Ok, now let's prove our Patience…

I told you in the beginning that we were going to test also a text-generation model (GPT2 based models).

I did everything the same, but I got very bad results… and I couldn't know it in advance. And here comes the patience: I couldn't learn without it.

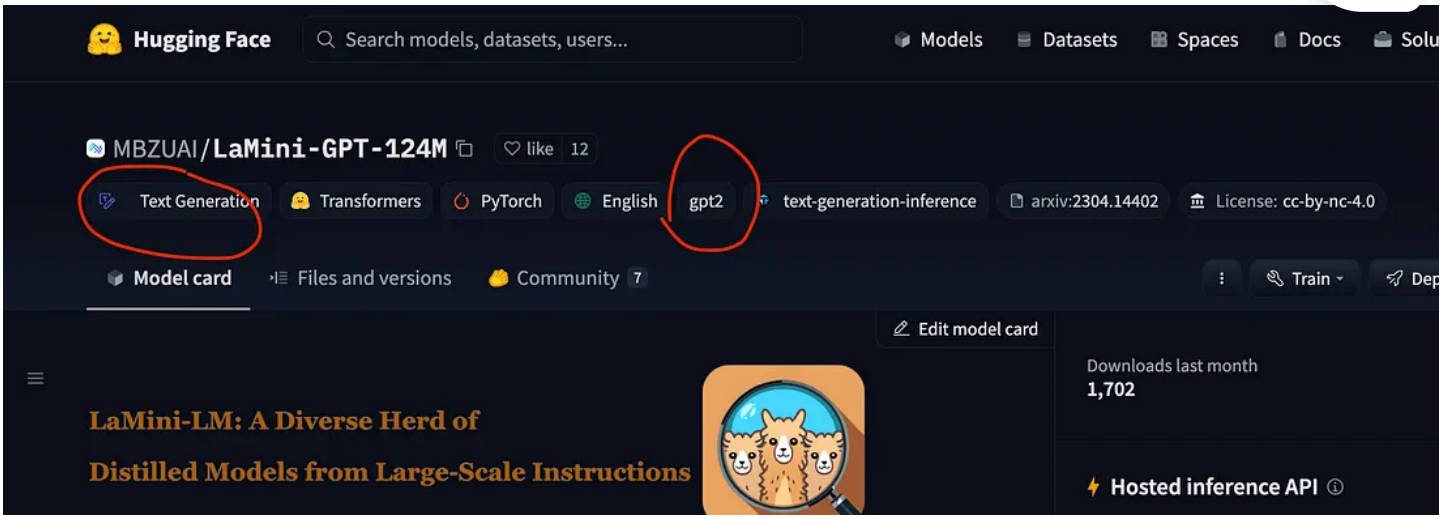
Let's take a LaMini model based on GPT2: go on the repository of the MBZUAI. Click on the Organization name next to the model



click on the organization — MBZUAI. It will open their repo

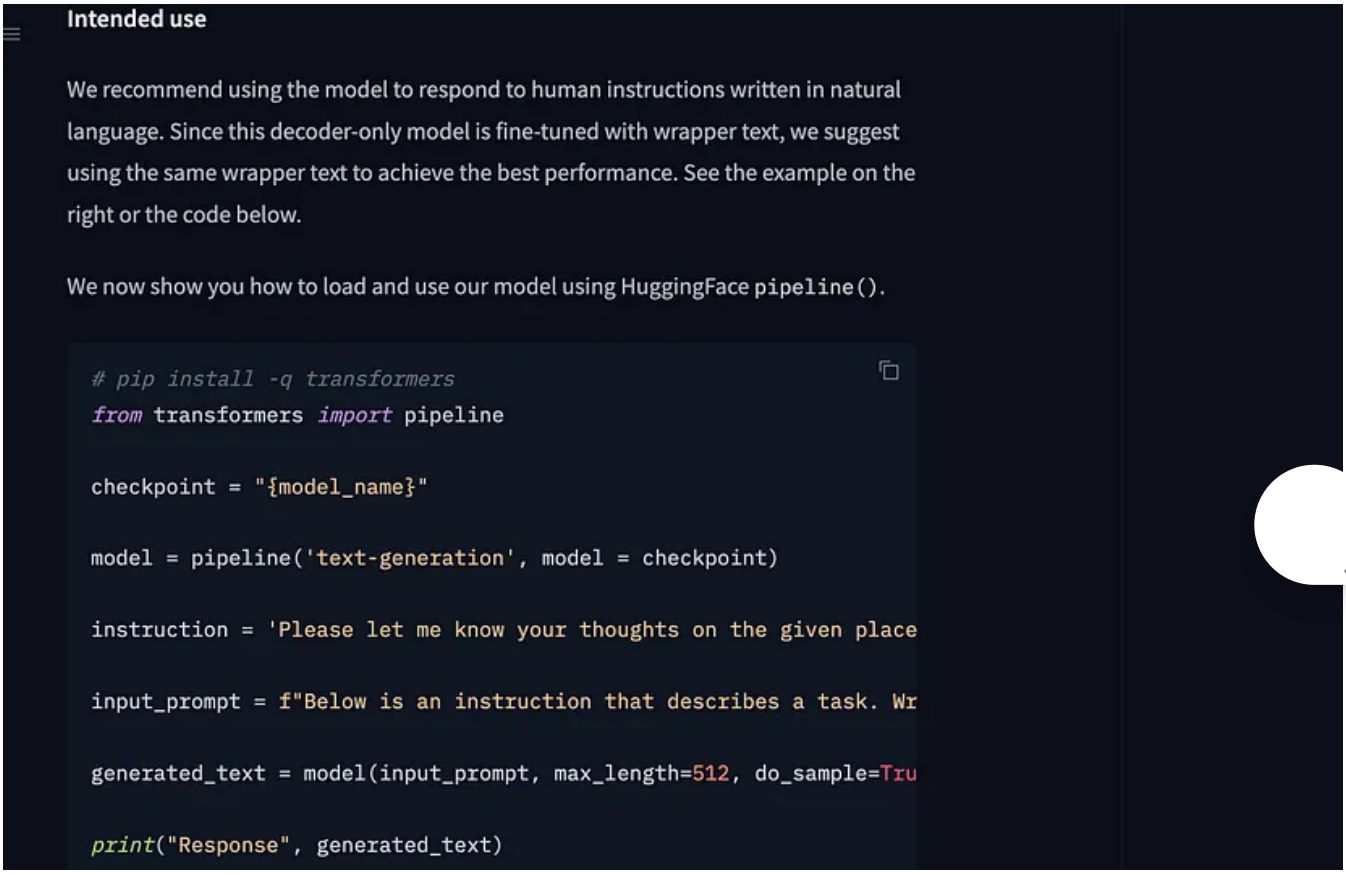
Scrolling down to the models you can see that we have some GPT models: we take the lightest one (to be sure our harware can take them)

click on MBZUAI/LaMini-GPT-124M



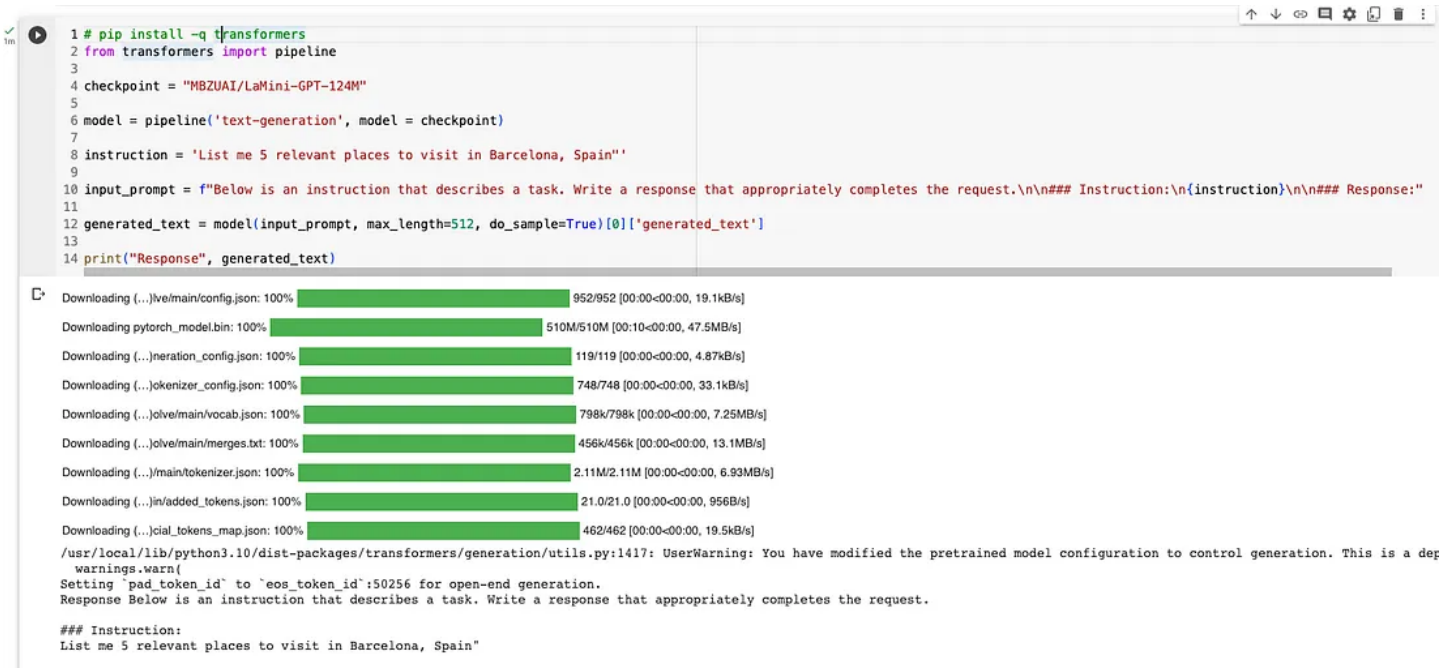
The model card is telling us that it is based on GPT2 and that it is trained on text-generation task

As we did before, scrolling down the model card page we can find the Use section. Also here we can copy/paste the code to run it in Google Colab.



Use the copy/paste icon and then paste it in a new cell in Google Colab

Remember to change {model name} with MBZUAI/LaMini-GPT-124M!!!



The model will be downloaded in the Colab Notebook and AI generation is printed

As you can see I tried to change a little the instruction, but again I got very bad results...

```
### Instruction:
List me 5 relevant places to visit in Barcelona.

### Response:1. Euskaltel Arena
2. Arc de Triomphe
3. Vicente Del Bosque
4. Marrakech
5. Vicente Carró
```

Bottom line: it is true that **we basically learned how to run any relatively small model from Hugging Face on our Notebook.** It is also true that not all the first attempts are good, and sometimes some models are better than others, but we cannot know until we test them ourselves.

As a rule of thumb if a model weight is 1Gb, it will require 2 times RAM to be loaded into the memory (you need 2 Gb of RAM to execute). If you have a model like the new famous meta-llama/Llama-2–7b that weights 13.5 Gb, you need 26 Gb of RAM… Not considering here any difference between having only a CPU and or a GPU.

a knowledge network… on Medium — generated with Leonardo.AI

You need a knowledge network

A knowledge network is a group of people who share, share and create new knowledge. They can capture and share information such as documents, stories,

metaphors, and metadata.

They are influenced by their environment, such as the management system of the group and surrounding culture. They are known for capturing, sharing and creating knowledge.

They are supported by tools such as information and communication tools as well as meeting time and rooms.

We need a knowledge network because we are learners. If you start to learn a complete new skill you cannot do anything alone: you need kind people that ca help you navigate in the territory.

Medium is one of these places: you have a lot of writers that share their knowledge in a way that anyone can learn. If you are a beginner there is something for you: if you are advanced level, there is something for you.

I also joined Medium because I was looking for a place with connection, good quality articles, and a forge of new ideas.

You can find the Notebook in my github repo

GitHub - fabiomatricardi/yourFirstAIgeneration: Repo of the code from the Medium article "Your...

Repo of the code from the Medium article "Your first AI generation" - GitHub - fabiomatricardi/yourFirstAIgeneration...

github.com

Hope you enjoyed the article. If this story provided value and you wish to show a little support, you could:

1. Clap a lot of times for this story
2. Highlight the parts more relevant to be remembered (it will be easier for you to find it later, and for me to write better articles)

3. Sign up for a Medium membership using my link — ($5/month to read unlimited Medium stories)
4. Follow me on Medium
5. Read my latest articles https://medium.com/@fabio.matricardi

Meantime you can check:

10 things to know before starting to work with Open source LLM — part 1

Learn the basics to start working with Open Source Large Language Models, and leverage the power of your own private AI

artificialcorner.com

12 things I wish I knew before starting to work with Hugging Face LLM

Insights and Tips for Navigating the Hugging Face LLM Landscape

artificialcorner.com

Artificial Intelligence | Plain English | Llm | Python | Hugging Face

Following

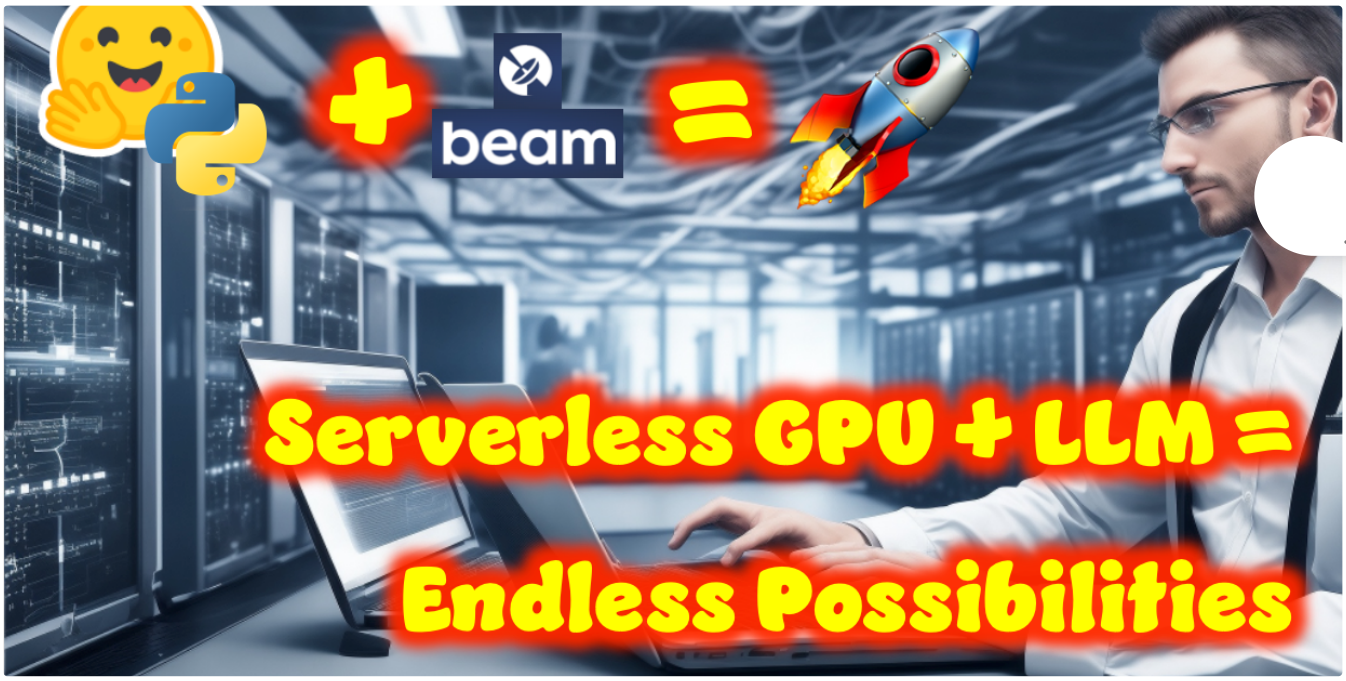
Written by Fabio Matricardi

2.7K Followers · Writer for Artificial Corner

passionate educator, curious industrial automation engineer. Learning Leadership and how to build my own AI. contact me at fabio.matricardi@gmail.com

---

More from Fabio Matricardi and Artificial Corner



Fabio Matricardi in Artificial Corner

Compute without Constraints: Serverless GPU + LLM = Endless Possibilities

Spark conversations through the cloud with serverless GPUs powering your most advanced Hugging Face large language models.

17 min read · Aug 29

280 1

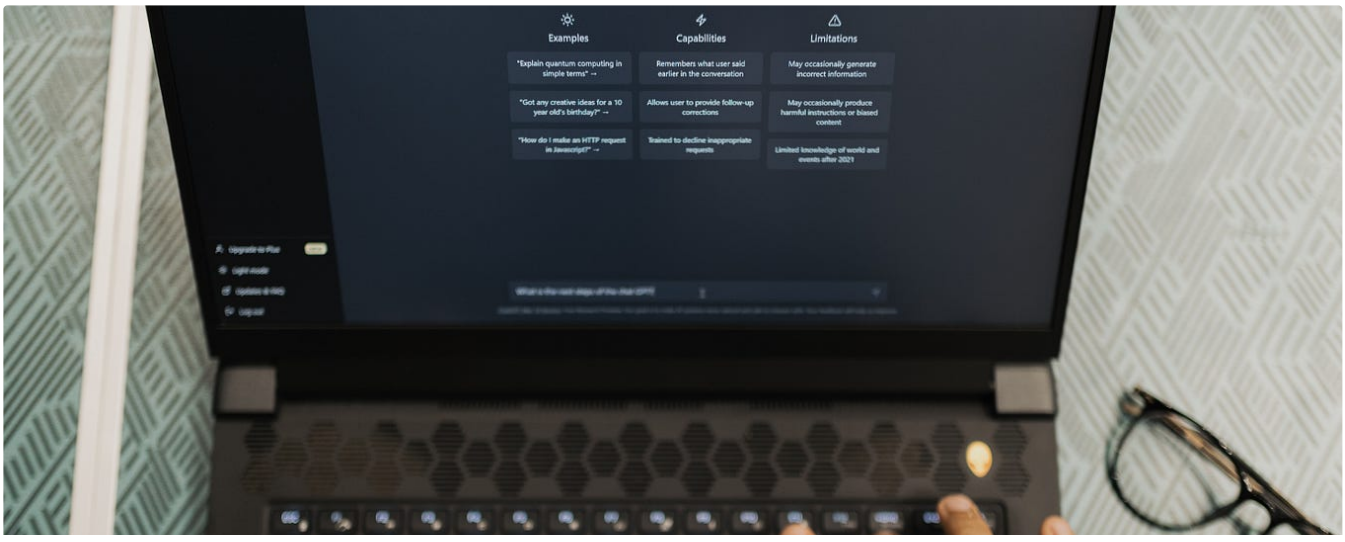
The PyCoach in Artificial Corner

My Honest Review of Some Paid AI Tools I've Used So Far

Here's why I canceled and kept some of these subscriptions so far.

· 6 min read · Aug 15

2.5K 36



Diana Dovgopol in Artificial Corner

I Used ChatGPT (At Work) for 6 Months. Here's How to 10X Your Productivity

ChatGPT can help automate the most boring parts of anyone's job.

· 7 min read · Jun 7

2K 28

Fabio Matricardi in Artificial Corner

Your Local LLM on your Network with FastAPI—part 2

Learn how to run your local FREE Hugging Face Language Model with Python, FastAPI and Streamlit.

· 10 min read · Aug 3

534 2

See all from Fabio Matricardi

See all from Artificial Corner

Recommended from Medium

AI TutorMaster in Level Up Coding

Falcon 180B: The new future of LLM's (Better Performance than ChatGPT 3.5)

Dive deep into the world's largest open language model, its capabilities, and how to harness its advanced configurations

· 8 min read · 4 days ago

183 1

Nick Hilton

The End of the Subscription Era is Coming

You’re overpaying for your porn (and journalism)

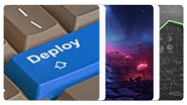
10 min read · Aug 30

9.2K 172

Lists

ChatGPT

21 stories · 151 saves



Predictive Modeling w/ Python

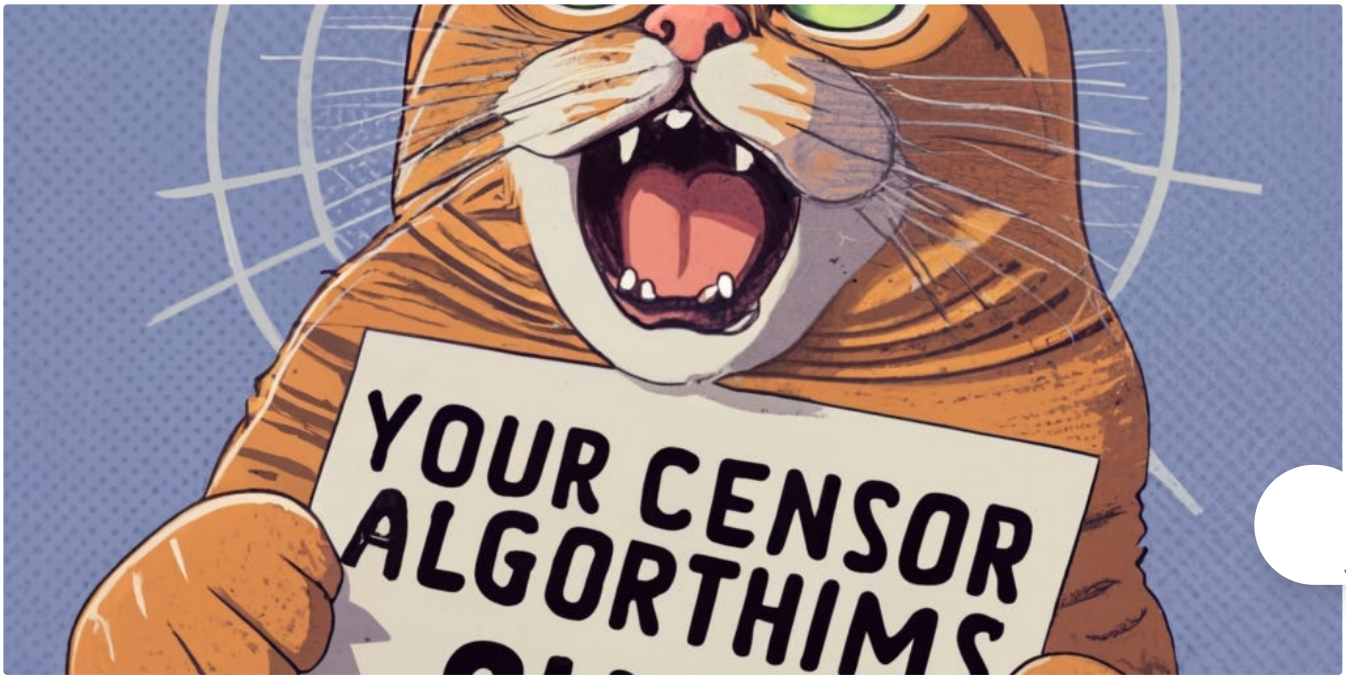
20 stories · 380 saves

ChatGPT prompts

24 stories · 372 saves

AI Regulation

6 stories · 119 saves



SM Raiyyan in ILLUMINATION

This New AI Generates Images with Text

GPT-4: 22 Latest AI Tools and News for Top 1% AI Fanatics

· 4 min read · Sep 6

189 3

Edmond Yip in Generative AI

17 Stable Diffusion SDXL style prompts to create stunning image

These prompts offer a wide range of creative possibilities across various artistic styles and genres

✨ · 6 min read · 5 days ago

201 2

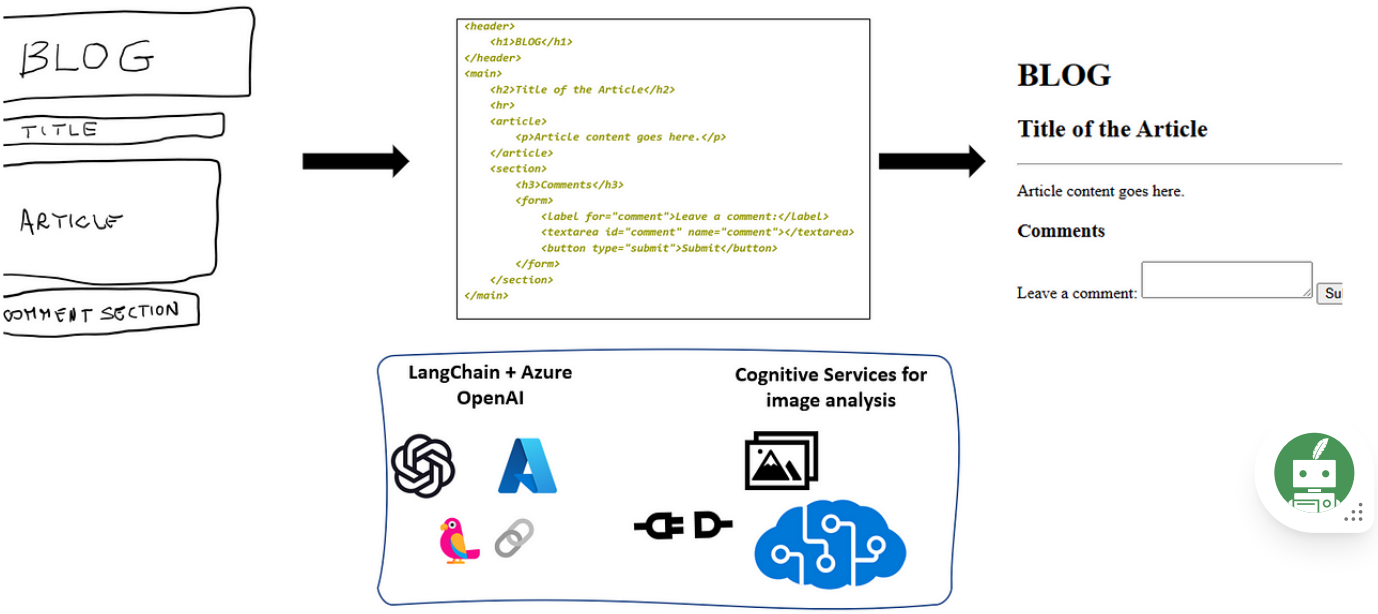
John Damask in Better Programming

Employee Search With OpenAI, Flowise, and LangChain

If you read my previous article, you know that I have a Bostonian version of ChatGPT in Slack called W'kid Smaaht ("wicked smart")

9 min read · Sep 6

221 6



Valentina Alto in Microsoft Azure

Generating applications from sketches with LLMs

An implementation with LangChain and GPT-3.5-turbo

· 7 min read · 6 days ago