

Open in app ↗



Search

✦ **Jump-start your best year yet:** Become a member and get 25% off the first year

Generating Summaries for Large Documents with Llama2 using Hugging Face and Langchain



Ankit · Follow

11 min read · Aug 27, 2023



Listen



Share

... More

Introduction

In my debut blog, I'm delving into the world of automatic summarization, a vital tool in our information-driven era. The introduction of Meta's LLama2 model has revolutionized this domain, making summarization more accessible and efficient than ever. In this tutorial, I'll unveil how LLama2, in tandem with Hugging Face and LangChain — a framework for creating applications using large language models — can swiftly generate concise summaries, even for substantial documents such as intricate patient reports. Let's embark on this exploration of the art of summarization, powered by cutting-edge technology.

Before You Begin: Access and Resources

To begin your journey into generating concise summaries, ensure you're equipped with the essential tools and resources. The synergy between Hugging Face, LLama2, and LangChain is a testament to the potent text processing capabilities available today. Here's your roadmap to getting started:

1. **Access Requests:** Start by requesting access from both Hugging Face and Meta, using the **same email ID** for both platforms. This ensures a seamless and integrated experience.
2. **Access Links:** Utilize the following access links:

- For Hugging Face: [meta-llama/Llama-2-7b-hf](https://huggingface.co/meta-llama/Llama-2-7b-hf)
- For Meta: meta.com/resources/models-and-libraries

Remember to log in with the same email ID used during your access requests.

3. Access Token: To interact with Hugging Face's models, you'll need an access token. Go to [Hugging Face's token page](#). Note down your token, as you'll require it in the upcoming steps.

Access Tokens

User Access Tokens

Access tokens programmatically authenticate your identity to the Hugging Face Hub, allowing applications to perform specific actions specified by the scope of permissions (read, write, or admin) granted. Visit [the documentation](#) to discover how to use them.



4. Hardware Considerations: Efficient text processing relies on powerful hardware. For example, if you're using Google Colab, consider utilizing a high-end processor like the A100 GPU. The hardware requirements can vary based on the size and complexity of your documents.

5. Downloading and Running Models Locally: You also have the option to download and run these models locally. In the email from Meta where you receive access, there will be instructions on how to download the models. If you have a good GPU instance, you can run them locally with ease. If you'd like me to create a blog post on this topic, let me know. I'd be happy to write an article describing the whole process.

Initiating the Summarization Quest: Hugging Face, Llama2, and Langchain

Crafting concise summaries for extensive documents is within reach through the synergy of Hugging Face, Llama2, and Langchain. Let's get started:

Install Essential Packages:

```
!pip install -q transformers langchain accelerate
```

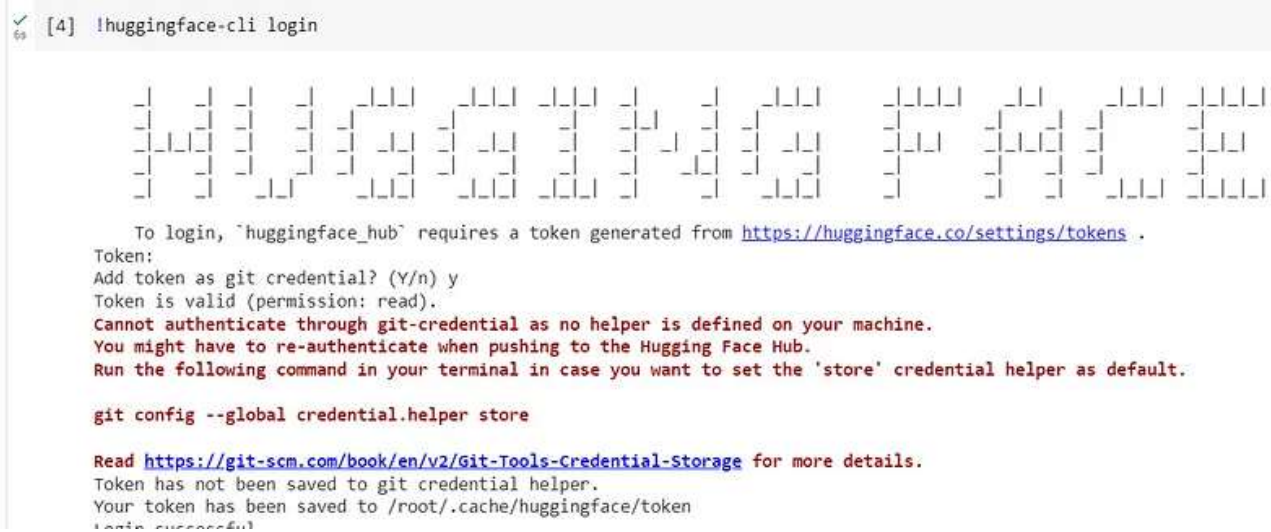
Hugging Face Cli Login

In your Google Colab notebook or Windows cmd, enter:

```
huggingface-cli login
```

`huggingface-cli login` command is crucial for authenticating your Hugging Face account, granting you access to a world of pre-trained models.

Input the token you generated earlier. After that, press 'y'

A terminal window showing the execution of the 'huggingface-cli login' command. The output displays the 'HUGGINGFACE' logo in ASCII art, followed by instructions to use a token from 'https://huggingface.co/settings/tokens'. It prompts for a token, asks if it should be added as a git credential (answered 'y'), and confirms the token is valid with 'read' permission. It then shows an error: 'Cannot authenticate through git-credential as no helper is defined on your machine.' and provides instructions to run 'git config --global credential.helper store' to resolve the issue. Finally, it shows the token being saved to '/root/.cache/huggingface/token' and confirms the login was successful.

```
[4] !huggingface-cli login

HUGGINGFACE

To login, 'huggingface_hub' requires a token generated from https://huggingface.co/settings/tokens .
Token:
Add token as git credential? (Y/n) y
Token is valid (permission: read).
Cannot authenticate through git-credential as no helper is defined on your machine.
You might have to re-authenticate when pushing to the Hugging Face Hub.
Run the following command in your terminal in case you want to set the 'store' credential helper as default.

git config --global credential.helper store

Read https://git-scm.com/book/en/v2/Git-Tools-Credential-Storage for more details.
Token has not been saved to git credential helper.
Your token has been saved to /root/.cache/huggingface/token
Login successful
```

Import Packages:

To kickstart our journey into generating insightful summaries for large documents, we need to ensure we have the right tools at our disposal. Begin by importing the necessary packages that form the foundation of our solution:

```
import torch
import transformers
```

```
from transformers import AutoTokenizer  
from langchain import LLMChain, HuggingFacePipeline, PromptTemplate
```

Model and Tokenization:

At the heart of our summarization process lies the selection of a powerful language model and efficient tokenization. Let's set the stage by configuring the model and tokenizer:

```
model = "meta-llama/Llama-2-7b-chat-hf"  
tokenizer = AutoTokenizer.from_pretrained(model)
```

In the code above, we pick the meta-llama/Llama-2-7b-chat-hf model. This model, used with Hugging Face's HuggingFacePipeline, is key to our summarization work. The tokenizer, made from the model, turns text into a format the model can handle well. With these parts ready, we're set to unleash AI-powered summarization.

Choosing the Right Model: Exploring Llama2 Variants

Our pursuit of powerful summaries leads to the meta-llama/Llama-2-7b-chat-hf model — a Llama2 version with 7 billion parameters. However, the Llama2 landscape is vast. The [Meta-Llama Model Repository](#) hosts various versions, each with unique parameters. For those craving top-notch summarization, models with more parameters hold appeal, promising refined summaries. Yet, remember, more parameters mean more computing power required. Your decision hinges on your needs. Explore the repository, assess Llama2 options, and consider document complexity, hardware, and accuracy. Choosing thoughtfully ensures your model matches your goals, balancing performance and efficiency.

Creating the Summarization Pipeline

At the core of our summarization method is a well-built pipeline that combines AI skills with language expertise. Let's examine the parts that come together to shape this process:

```
pipeline = transformers.pipeline(  
    "text-generation",  
    model=model,  
    tokenizer=tokenizer,  
    torch_dtype=torch.bfloat16,  
    trust_remote_code=True,  
    device_map="auto",  
    max_length=3000,  
    do_sample=True,  
    top_k=10,  
    num_return_sequences=1,  
    eos_token_id=tokenizer.eos_token_id  
)
```

In the code above, we create a smart pipeline using Hugging Face's Transformers library. This pipeline is like a conductor for summarization. The chosen model, tokenizer, and different settings — like `max_length` and `eos_token_id` — work together for this AI magic.

In the code snippet, notice the **`max_length` set to 3000**. It helps prevent issues if the document is too long for the AI. Adjusting this `max_length` according to your document's length keeps the AI's insights clear and avoids errors.

By setting the `max_length` parameter right, you harmonize the document's length with the AI's brevity. This creates a smooth summarization process without errors.

Fine-Tuning with Temperature: Customizing the Output

To infuse our generated summaries with a touch of finesse, we turn our attention to the parameter `'temperature'`. This nuanced parameter impacts the diversity and randomness of our AI-generated outputs.

```
llm = HuggingFacePipeline(pipeline = pipeline, model_kwargs = {'temperature':0})
```

In the code above, we use the `HuggingFacePipeline` to shape our summarization process. By adding `model_kwargs`, we tweak the outcome with the temperature

setting. A `temperature` of 0 gives direct summaries, while higher values add some randomness.

When you set the `temperature`, think about your readers and situation. For formal documents, a lower value might be fitting, while a higher value could be engaging for creative pieces. This customization lets you make AI-generated summaries match your style and vibe just right.

Crafting a Guiding Template: Summarization Made Seamless

To channel the power of AI into generating succinct and comprehensive summaries, we introduce the concept of a guiding template. This template serves as a blueprint to direct the summarization process and ensure the encapsulation of key points.

```
template = """
    Write a summary of the following text delimited by triple backticks
    Return your response which covers the key points of the text.
    ```{text}```
 SUMMARY:
 """
```

The template structure is ingenious in its simplicity. It first provides clear instructions, encapsulated within triple backticks, signaling the text you intend to summarize. Your role is to craft a response that distills the essence of the content. The concluding touch is the dedicated "SUMMARY:" section that punctuates your generated summaries.

This template not only streamlines the summarization process but also provides a framework to encapsulate the document's core points. It serves as your artistic palette, allowing you to harness AI's potential while weaving your human touch into the fabric of each summary.

## The Power of a Guiding Template

In the world of AI summarization, your template isn't just a blueprint — it's the captain's wheel that guides the AI's insights. Crafting a precise template is key. Random choices won't yield aligned results. For deeper insights into effective



template design, explore [Sophia Yang, Ph.D.](#)'s article on [Best Practices in Prompt Engineering](#). It's a guiding light to optimize AI summaries through strategic template construction.

The right template seamlessly merges AI potential with your human touch, creating summaries that engage and captivate.

## Elevating Summarization with Langchain

Once you've established the core pipeline, it's time to elevate your summarization capabilities with Langchain. With the prowess of Langchain, generating insightful summaries becomes an attainable goal. Utilize `PromptTemplate` to structure your summarization process and `LLMChain` to seamlessly connect Langchain with your pipeline. Here's how you can achieve this synergy: Now that our groundwork is laid, we venture into the world of prompts—the vital link between human intent and AI-generated summaries.

```
prompt = PromptTemplate(template=template, input_variables=["text"])
llm_chain = LLMChain(prompt=prompt, llm=llm)
```

The `PromptTemplate` shapes the dialogue, adapting dynamically to your input. Paired with the `LLMChain`, this synergy fuses human creativity with AI precision, forging summaries that captivate.

Together, these components are the key, unlocking a harmonious partnership between your vision and the AI's prowess—a partnership that crafts summaries resonating seamlessly with your audience.

## The Challenge: Summarizing a 4000-Word Patient Report

Our quest to showcase AI-powered summarization led us to a unique challenge: requesting ChatGPT to generate an extensive 4000-word patient report. Below is the detailed patient report, meticulously crafted by ChatGPT:

```
text = ""
Patient Name: John Doe
Date of Birth: January 15, 1975
```

Gender: Male

Medical Record Number: 123456789

Date of Assessment: August 18, 2023

#### I. Chief Complaint:

The patient presents with complaints of persistent fatigue, unexplained weight

#### II. Medical History:

The patient has a history of hypertension managed with medication for the past

#### III. Review of Systems:

General: The patient reports fatigue, unexplained weight loss of approximately

Gastrointestinal: The patient experiences intermittent abdominal pain, predomin

Cardiovascular: The patient's blood pressure has been well controlled with medi

Respiratory: The patient denies cough, wheezing, or shortness of breath.

Musculoskeletal: No significant joint pain or limitations in mobility reported.

Neurological: The patient denies headaches, dizziness, or changes in vision.

Psychological: The patient mentions occasional stress due to work-related facto

#### IV. Physical Examination:

Vital Signs: Blood pressure is 130/80 mmHg, heart rate is 78 beats per minute,

General: The patient appears fatigued but alert and oriented to person, place,

Abdominal Examination: There is tenderness on palpation in the right upper quad

Cardiovascular Examination: Regular rate and rhythm with no murmurs or abnormal

Respiratory Examination: Clear breath sounds bilaterally, no wheezing or crackl

Neurological Examination: No focal neurological deficits observed.

#### V. Diagnostic Investigations:

Complete Blood Count (CBC): Within normal limits.

Comprehensive Metabolic Panel (CMP): Slight decrease in albumin levels.

Liver Function Tests (LFTs): Mild elevation in liver enzymes (AST and ALT).

Abdominal Ultrasound: No evidence of gallstones or other abnormalities. Liver a

CT Scan of the Abdomen: Reveals a mass in the liver, approximately 5 cm in diam

#### VI. Assessment and Plan:

Based on the patient's symptoms, physical examination, and diagnostic investiga

Assessment:



Unexplained weight loss and fatigue.  
Right upper quadrant abdominal pain.  
Elevated liver enzymes and an enlarging liver mass.  
Plan:

Further Evaluation: Given the presence of an enlarging liver mass and elevated  
Oncology Consultation: Given the possibility of malignancy, an oncology consult  
Symptom Management: The patient's abdominal pain will be managed with pain relief  
Nutritional Support: The patient's decreased appetite and weight loss will be addressed  
Psychological Support: Given the patient's stress and anxiety related to his symptoms  
VII. Follow-Up:

The patient is scheduled for a follow-up appointment in two weeks to discuss the

VIII. Prognosis and Discussion:

The presence of an enlarging liver mass raises concerns about potential malignancy.

IX. Patient Education:

The patient and his family will receive comprehensive education about the diagnosis.

X. Conclusion:

This patient report outlines the comprehensive health assessment of John Doe, a 55-year-old male with a long history of hypertension and type 2 diabetes. The patient presents with symptoms of abdominal pain, weight loss, and fatigue, leading to the discovery of a liver mass. Further evaluation and management are required.

This comprehensive report mirrors the complexity of real-world medical assessments. It's the ideal input document for our summarization experiment, demonstrating how AI can distill vast amounts of information while retaining the crucial details.

As we dive into the experiment, observe how AI harnesses its summarization prowess to transform lengthy narratives into succinct insights.

## Unveiling AI-Generated Summaries

With our input document in hand, it's time to witness the magic of AI-driven summarization in action.

```
print(llm_chain.run(text))
```

The code snippet above triggers the AI summarization process using the `llm_chain` we meticulously assembled. As the AI combs through the extensive patient report, it condenses the content into a concise summary that captures the essence of the narrative.

## 🚀 Finally, the Generated Summary 🚀

At last, we arrive at the culmination of our exploration — the generated summary derived from the patient report:



The screenshot shows a Jupyter Notebook interface with a file named 'summarization\_using\_llama2.ipynb'. The notebook contains a code cell with the following text:

```
print(llm_chain.run(text))
```

The output of the code cell is a concise summary of the patient report:

```
John Doe, a 48-year-old male patient, presented with persistent fatigue, unexplained weight loss, and intermittent abdominal pain over the past few months. His medical h
```

*“John Doe, a 48-year-old male patient, presented with persistent fatigue, unexplained weight loss, and intermittent abdominal pain over the past few months. His medical history includes hypertension, appendectomy, and hernia repair surgery. The patient’s vital signs were normal, and his physical examination revealed tenderness in the right upper quadrant of the abdomen. Diagnostic investigations, including a complete blood count, comprehensive metabolic panel, liver function tests, abdominal ultrasound, and CT scan of the abdomen, revealed elevated liver enzymes and an enlarging liver mass. Based on the assessment and plan, the patient will be referred to a gastroenterologist for further evaluation and management, and an oncology consultation will be sought to determine the nature of the liver mass. The patient’s prognosis will be influenced by the nature of the liver mass and the success of treatment interventions.”*

This definitive AI-generated summary attests to the transformative power of Llama2, condensing intricate narratives into concise, informative insights. While compact, it encapsulates the vital components of the patient report, showcasing how AI-driven summarization bridges the divide between exhaustive documents and impactful takeaways.

## Prioritizing Data Privacy and Security

As you traverse the realm of text summarization, data privacy and security are paramount. Hugging Face acknowledges these concerns and offers safeguards for your data. For those concerned about data privacy, Hugging Face provides a Business Associate Addendum or GDPR data processing agreement through the Inference Endpoint enterprise plan. Discover more about their robust security standards [here](#). Additionally, for those who seek greater control over their data and processes, you can opt to download models and run them locally, thereby enhancing your control and independence while ensuring the utmost privacy and security. If you'd like me to create a blog post on this topic, let me know. I'd be happy to write an article describing the whole process.

## Conclusion

Thank you for joining me on this journey. Abstractive summarization using Llama2, Hugging Face, and Langchain offers an efficient way to swiftly comprehend lengthy documents. We've learned to seamlessly blend our insights with LLM capabilities to create impactful summaries. Dive in and begin your summarization adventures! If you found this article helpful, a thumbs-up would be much appreciated. Please feel free to share any feedback — your insights drive our ongoing evolution.

[Abstractive Summarization](#)[Llama 2](#)[Medical Documents](#)[Text Summarization](#)[Document](#)[Follow](#)

## Written by Ankit

29 Followers

As a Data Scientist with 5+ years in healthcare, I've expertly applied Machine Learning and NLP to drive transformative outcomes.

## Recommended from Medium



Gao Dalie (高達烈) in Level Up Coding

### Text Summarization Llama2: how to Use LLama2 with Langchain

In this Tutorial, I will guide you through how to use LLama2 with langchain for text summarization and named entity recognition using...

★ • 10 min read • Jul 27, 2023



216



2





Tushit Dave

## Llama2 and Text Summarization

Unlocking the Power of Llama2 for Local Multi-Document Summarization

9 min read · Sep 9, 2023



191



7



### Lists



#### data science and AI

39 stories · 44 saves



#### Natural Language Processing

1103 stories · 572 saves





Thomas Jaensch

## Summarize any text based document with 8 lines of Python and LlamaIndex

Use the power of LLMs to summarize or ask questions about any custom document without having to write model plumbing code, LangChain or...

4 min read · Sep 30, 2023



18



1





Murtuza Kazmi

## Using LLaMA 2.0, FAISS and LangChain for Question-Answering on Your Own Data

Over the past few weeks, I have been playing around with several large language models (LLMs) and exploring their potential with all sorts...

9 min read · Jul 24, 2023



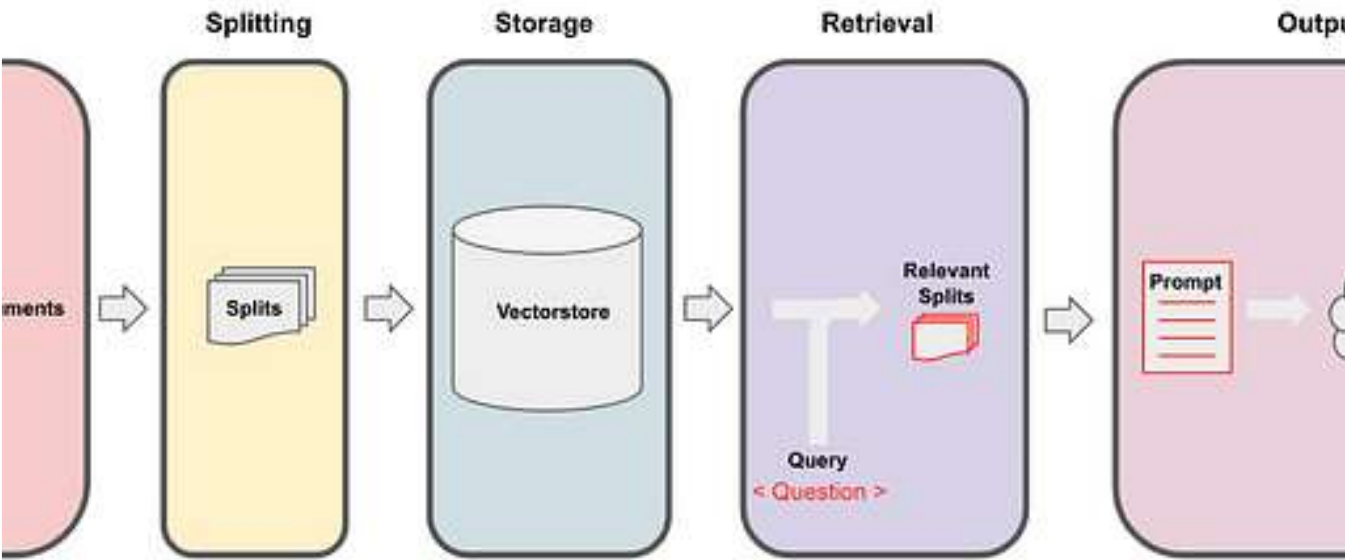
1.2K




15







 Onkar Mishra

## Using langchain for Question Answering on own data

Step-by-step guide to using langchain to chat with own data

23 min read · Aug 7, 2023

 1.1K

 13







 Johni Douglas Marangon

## How to summarize text with OpenAI and LangChain

This is the first of three posts about my recent study of summarization using Large Language Models—LLM.

7 min read · Aug 17, 2023

 85

 3





See more recommendations