Langchain Tutorial for the LOST

It's a strange time we are living in. All the exciting LLMs are coming, chatGPT is acting like know-it-all, there's debate floating around: Are we getting AGI?

If you are a **human**, and jutifiably feeling somewaht **overwhelmed**, I am just trying to help you get started on **Langchain**

Talk of the town: But what it is?

Langchain is just a framework that helps you connect your application with the LLM.

Do I need Langchain?

If you are just **playing around** with chatGPT and other LLMs, then NO! But if you are:

1. a developer
2. a researcher

then, YES ! Langchain is a super helpful tool !

Let's install langachain first

In [1]:
```
# uncomment:
#!pip install langchain
```

In [2]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
import langchain
```

In [4]:
```
langchain.__version__
```

Out[4]: '0.1.0'

NB: A lot can depend on the version, since langchain is a work on progress.

Prompting

We all kinda know, much of LLM's magic lies in the way we prompt. Langchain helps you to prompt in a smart way.

As an LLM, we will use openai gpt-3.5 turbo. You need to get the openai key for this. Let me know if you need any

In [4]:
```
import os
os.environ["OPENAI_API_KEY"] = "**********************************************"
```

Ofcourse, I will not reveal my key. Get the key and replace the asterisks with it.

In [6]:
```
from langchain.llms import OpenAI
```

In [7]:
```
llm = OpenAI(temperature=0.6,  model="gpt-3.5-turbo-instruct")
```

```
/home/sadat/.local/lib/python3.8/site-packages/langchain_core/_api/deprecation.py:117: La
ngChainDeprecationWarning: The class `langchain_community.llms.openai.OpenAI` was depreca
ted in langchain-community 0.0.10 and will be removed in 0.2.0. An updated version of the
class exists in the langchain-openai package and should be used instead. To use it run `p
ip install -U langchain-openai` and import as `from langchain_openai import OpenAI`.
  warn_deprecated(
```

Let's try a simple prompt

In [8]:
```
name = llm("I have expertise on t-shirt painting. Suggest a cool startup idea in very brie
print(name)
```

```
/home/sadat/.local/lib/python3.8/site-packages/langchain_core/_api/deprecation.py:117: La
ngChainDeprecationWarning: The function `__call__` was deprecated in LangChain 0.1.7 and
will be removed in 0.2.0. Use invoke instead.
  warn_deprecated(


A personalized t-shirt subscription service that allows customers to choose from a variet
y of unique designs and have them hand-painted onto high-quality t-shirts by local artist
s. Customers can also submit their own designs to be painted, making each t-shirt truly o
ne-of-a-kind. This startup would offer a fun and creative way for people to express thems
elves through fashion while supporting local artists.
```

Now, we can keep on exploring this fun experiment with different startup ideas based on your expertise like software engineering, maths, chemistry, salesperson etc. But what we want is, a smart way of doing this

In [9]:
```
from langchain.prompts import PromptTemplate
```

In [10]:
```
prompt_template_skill = PromptTemplate(
    input_variables = ["skill"],
    template = "I have expertise on {skill}. Suggest a cool startup idea."
)
```

In [11]:
```
# That's what the prompt looks like
prompt_template_skill.format(skill="data science and machine learning")
```

Out[11]:
```
'I have expertise on data science and machine learning. Suggest a cool startup idea.'
```

In [12]:
```
# A glimpse of chain ;)
from langchain.chains import LLMChain
```

In [13]:
```
prompt_template_skill_chain =  LLMChain(llm=llm, prompt=prompt_template_skill)
prompt_template_skill_chain.run("data science and machine learning")
```

```
/home/sadat/.local/lib/python3.8/site-packages/langchain_core/_api/deprecation.py:117: La
ngChainDeprecationWarning: The function `run` was deprecated in LangChain 0.1.0 and will
be removed in 0.2.0. Use invoke instead.
  warn_deprecated(
```

Out[13]:
```
"\n\nA personalized virtual wardrobe assistant that uses machine learning to suggest outf
it combinations based on the user's style preferences, weather, and occasion. The assista
nt would also keep track of the user's wardrobe and suggest items to purchase to fill any
gaps in their wardrobe. It could also offer styling tips and inspiration from fashion inf
luencers. This idea combines both data science and machine learning to provide a unique a
nd personalized fashion experience for users."
```

In [14]:
```
# Let's try another fun one
prompt_template_skill_chain.run("sleeping and eating")
```

Out[14]:
```
"\n\nA sleep and nutrition tracking app that uses artificial intelligence to create perso
nalized sleep and meal plans for individuals. The app would track sleep patterns and eati
ng habits, and provide recommendations for improving both based on the user's goals and p
references. It could also offer features such as guided meditations for better sleep and
healthy recipe suggestions. The app could partner with fitness trackers and meal delivery
services to offer a comprehensive wellness experience."
```

You can play around with different fun prompting strategy. Let me know what else you tried :D

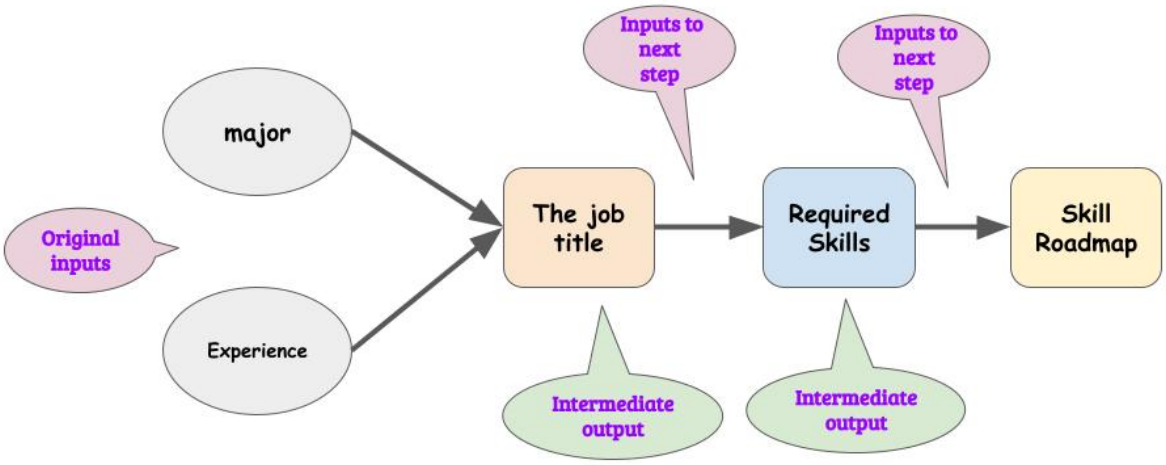
Chains

Now, this is the part I love most about Langchain. You can literally create a chain of action using LLMs. Instead of providing a long step-by-step chain of thoughts instruction in one single prompt, you can distribute it across a chain, making sure one action is done before taking another action

In [15]:
```
from langchain.chains import SequentialChain
```

Say, we are brainstorming our career option (unless already doomed by AI lol). Now, I want to build a model which can take my major and experience as input, and find the suitable job title. I also want it to name the required skills I need from the title. Finally, I want my skill roadmap based on the skills I need.

Yes, life is beautiful with Sequential Chain



In [16]:
```
# prompt template 1: Ask the question of job title
job_title_prompt = PromptTemplate.from_template(
    "Assume the role of a career coach. Suggest me the title of the single best job, based
    Also consider the experience level of {experience}\n"
)
# chain JT: input= major and experience and output= job_title
chain_JT = LLMChain(llm=llm, prompt=job_title_prompt,
                    output_key="job_title"
                   )
```

In [17]:
```
# prompt template 2: Ask the question of Required skills
reqd_skills_prompt = PromptTemplate.from_template(
    "What are the name of top 3 required skills for the job title: {job_title}. Donot exte
)
#  chain_RS: input= job_title and output= reqd_skills
chain_RS = LLMChain(llm=llm, prompt=reqd_skills_prompt,
                    output_key="reqd_skills"
                   )
```

In [18]:
```
# prompt template 3: Ask the ways of roadmap for the skills
skill_roadmap_prompt = PromptTemplate.from_template(
    "Tell me the very brief roadmap of getting the following skills: {reqd_skills}\n"
)
#  chain_SR: input= reqd_skills and output= skill_roadmap
chain_SR = LLMChain(llm=llm, prompt=skill_roadmap_prompt,
                    output_key="skill_roadmap"
                   )
```

In [19]:
```
# overall_chain: input= {major} and {experience}
# and output= Job title, required skills, skill roadmap
overall_chain = SequentialChain(
    chains=[chain_JT, chain_RS, chain_SR],
    input_variables=["major", "experience"],
    output_variables=["job_title", "reqd_skills","skill_roadmap"],
    verbose=True
)
```

Finally, we just need to tie the chains together to build a overall chain

In [20]:
```
Response = overall_chain({"major":"Data Scientist", "experience":"5 years"})
```

```
/home/sadat/.local/lib/python3.8/site-packages/langchain_core/_api/deprecation.py:117: La
ngChainDeprecationWarning: The function `__call__` was deprecated in LangChain 0.1.0 and
will be removed in 0.2.0. Use invoke instead.
  warn_deprecated(


> Entering new SequentialChain chain...

> Finished chain.
```

In [21]:
```
print(Response["job_title"])
```

```
"Senior Data Scientist: Maximizing Impact through Advanced Analytics and Strategic Insigh
ts"
```

In [22]:
```
print(Response["reqd_skills"])
```

```
1. Advanced Analytics
2. Strategic Insights
3. Data Science
```

In [23]:
```
print(Response["skill_roadmap"])
```

```
1. Advanced Analytics:
- Start by gaining a strong foundation in statistics, mathematics, and programming.
- Familiarize yourself with data analysis tools such as Excel, SQL, and Python.
- Learn advanced analytical techniques such as regression analysis, predictive modeling,
and data mining.
- Gain hands-on experience by working on real-world projects or participating in data ana
lytics competitions.
- Stay updated with the latest trends and developments in the field through online course
s, workshops, and conferences.

2. Strategic Insights:
- Develop a strong understanding of business strategy and decision-making processes.
- Familiarize yourself with data analysis tools and techniques, as well as market researc
h methods.
- Gain experience in synthesizing and interpreting data to provide actionable insights.
- Learn how to effectively communicate insights to stakeholders through presentations and
reports.
- Develop a strategic mindset and continuously seek opportunities to apply your skills in
a business context.

3. Data Science:
- Start by building a strong foundation in statistics, mathematics, and computer science.
- Learn programming languages such as Python and R, as well as data manipulation and visu
alization tools.
- Gain knowledge of machine learning algorithms and techniques.
- Familiarize yourself with big data technologies such as Hadoop and Spark.
- Gain hands-on experience through data science projects and internships
```

You can play around with a lot interesting and complex chain structures. What are some fun ideas?

Agent

Agents are pretty much the most revolutionary concepts of LLMs, that extends the functionalities of the chain concepts. It is where you connect the *external* knowledge source with your LLMs capability

Let's start with an example. We ask our LLM ofa knowledge that is after their release date (2021)

In [39]:
```
response = llm("When did Matthew Perry die and how?")
print(response)
```

```
Matthew Perry is still alive as of 2021. He was born on August 19, 1969, and is 51 years
old. There have been no reports of his death.
```

Let's bring the magic of agents now

In [42]:
```
## You may have to install wikipedia

#!pip install wikipedia
```

In [43]:
```
from langchain.agents import AgentType, initialize_agent, load_tools
from langchain.llms import OpenAI
```

In [48]:
```
tools = load_tools(["wikipedia"], llm=llm)

agent = initialize_agent(
    tools,
    llm,
    agent = AgentType.ZERO_SHOT_REACT_DESCRIPTION,
    verbose= False
)
```

In [49]:
```
agent.run("When did Matthew Perry die and how?")
```

Out[49]:
```
'Matthew Perry (the actor) died on October 28, 2023. The cause of his death is not specif
ied. Matthew Calbraith Perry (the naval officer) died on March 4, 1858.'
```

Voila !

In [51]:
```
from langchain.tools import BaseTool, StructuredTool, Tool, tool, DuckDuckGoSearchRun
from langchain import LLMMathChain
```

In [52]:
```

search = DuckDuckGoSearchRun()

search_tool = Tool.from_function(
    func=search.run,
    name="Search",
    description="useful for when you need to search the internet for information"
)

llm_math_chain = LLMMathChain(llm=llm, verbose=True)

math_tool = Tool.from_function(
    func=llm_math_chain.run,
    name="Calculator",
    description="Useful for when you are asked to perform math calculations"
)

tools = [search_tool, math_tool]
```

In [64]:

```
agent2 = initialize_agent(
    tools=tools,
    llm=llm,
    agent = AgentType.ZERO_SHOT_REACT_DESCRIPTION,
    verbose= True
)
```

In [65]:
```
agent2.run("What is the average income of US families? Also, multiply it by 80")
```

```
> Entering new AgentExecutor chain...
 I need to find the average income of US families and then multiply it by 80.
Action: Search
Action Input: "average income of US families"
Observation: Introduction This report presents estimates on income, earnings, and inequal
ity in the United States for calendar year 2022, based on information collected in the 20
23 and earlier Current Population Survey Annual Social and Economic Supplements (CPS ASE
C) conducted by the Census Bureau. Median annual family income in Canada 2020, by provinc
e ; ... Percentage distribution of household income in the United States in 2022 [Graph],
US Census Bureau, September 12, 2023. [Online]. The median family income in the United St
ates grew to 92,750 U.S. dollars in 2022, an increase on the previous year. Family income
is the total income earned by all family members who have been ... U.S. Census Bureau, Me
dian Family Income in the United States [MEFAINUSA646N], retrieved from FRED, Federal Res
erve Bank of St. Louis; https://fred.stlouisfed.org/series/MEFAINUSA646N, (https://fred.s
tlouisfed.org/series/MEFAINUSA646N,) January 11, 2024. RELEASE TABLES Income and Poverty
in the United States Personal and Family Income by Census Region, Annual RELATED DATA AND
CONTENT A further 19.3 percent of families in the United States had an annual income betw
een 100,000 and 149,999 U.S. dollars. ... Median annual family income in Canada 2020, by
earning situation ...
Thought: I should look for the median annual family income in the United States.
Action: Search
Action Input: "median annual family income United States"
Observation: This report presents estimates on income, earnings, and inequality in the Un
ited States for calendar year 2022, based on information collected in the 2023 and earlie
r Current Population Survey Annual Social and Economic Supplements (CPS ASEC) conducted b
y the Census Bureau. 2022: 92,750 (+ more) Updated: Sep 12, 2023 Units: Current Dollars,
Not Seasonally Adjusted Frequency: Annual 1Y | 5Y | 10Y | Max to Edit Graph Share Links A
ccount Tools NOTES Source: U.S. Census Bureau Release: Income and Poverty in the United S
tates Units: Current Dollars, Not Seasonally Adjusted Frequency: Annual Updated on Sept.
13 with the latest U.S. Census Bureau data. The national median household income is $74,5
80, according to the most recent U.S. Census Bureau Current Population Survey data... U.
S. household income distribution 2022 Published by Statista Research Department , Nov 3,
2023 In 2022, just over 50 percent of Americans had an annual household income that was l
ess... For an indication of the comparability of medians calculated using Pareto interpol
ation with medians calculated using linear interpolation, see Series P-60, Number 114, Mo
ney Income in 1976 of Families and Persons in the United States (www2.census.gov/prod2/po
pscan/p60-114.pdf). Suggested Citation:
Thought: I should use the median annual family income of $92,750.
Action: Calculator
Action Input: 92750 * 80

> Entering new LLMMathChain chain...
92750 * 80```text
92750 * 80
```
...numexpr.evaluate("92750 * 80")...

Answer: 7420000
> Finished chain.

Observation: Answer: 7420000
Thought: I now know the final answer.
Final Answer: The average income of US families is $92,750 and when multiplied by 80, it
is $7,420,000.

> Finished chain.
```

Out[65]:
```
'The average income of US families is $92,750 and when multiplied by 80, it is $7,420,00
0.'
```
```

You can see, our agent decides to search for the income information first, and then invoke the math chain to perform the multiplication

There are many different agents that can be extremely useful. Look it up, and play with it.

Memory

Let's break the news: **chatGPT is NOT an LLM**, rather it is an application, where one or more stellar LLM apis are tied with. LLMs are stateless, they donot have any memory. In fact, same goes with LLMchains

Let me prove that to you

In [66]:
```
llm("Can you tell me who was the president of USA in 1996? ")
```

Out[66]:
```
'\n\nThe president of the USA in 1996 was Bill Clinton.'
```

In [67]:
```
llm("Which party was in power back then? ")
```

Out[67]:
```
'\n\nIt is not specified which country or time period is being referred to, so it is impo
ssible to determine which party was in power. '
```

Ergo, it doesn't remember what was the topic we talked about !

So, Let's introduce memory.

In [137]:
```
from langchain.chains import ConversationChain
```

In [138]:
```
convo = ConversationChain(llm=llm)
```

In [143]:
```
prompt_template_president = PromptTemplate(
    input_variables = [],
    template = "Can you tell me who was the president of USA in 1996?"
)

prompt_template_party = PromptTemplate(
    input_variables = [],
    template = "Which party was in power back then?"
)
```

In [144]:
```
convo.run(prompt_template_president.format())
```

Out[144]:
```
' In 1996, the president of the USA was Bill Clinton. He was the 42nd president of the Un
ited States, serving from 1993 to 2001. He was a member of the Democratic Party and is kn
own for his economic policies, including the North American Free Trade Agreement and the
Balanced Budget Act of 1997.'
```

In [145]:
```
convo.run(prompt_template_party.format())
```

Out[145]:
```
" The Democratic Party was in power during Bill Clinton's presidency in 1996. The Democra
tic Party is one of the two major political parties in the United States, along with the
Republican Party. They have historically supported social welfare programs and progressiv
e policies."
```

Pretty cool, isn't it?

In [147]:
```
## Here's the memory content
convo.memory
```

Out[147]:
```
ConversationBufferMemory(chat_memory=ChatMessageHistory(messages=[HumanMessage(content='C
an you tell me who was the president of USA in 1996?'), AIMessage(content=' In 1996, the
president of the USA was Bill Clinton. He was the 42nd president of the United States, se
rving from 1993 to 2001. He was a member of the Democratic Party and is known for his eco
nomic policies, including the North American Free Trade Agreement and the Balanced Budget
Act of 1997.'), HumanMessage(content='Which party was in power back then?'), AIMessage(co
ntent=" The Democratic Party was in power during Bill Clinton's presidency in 1996. The D
emocratic Party is one of the two major political parties in the United States, along wit
h the Republican Party. They have historically supported social welfare programs and prog
ressive policies.")]))
```

In [149]:
```
print(convo.memory.buffer)
```

```
Human: Can you tell me who was the president of USA in 1996?
AI:  In 1996, the president of the USA was Bill Clinton. He was the 42nd president of the
United States, serving from 1993 to 2001. He was a member of the Democratic Party and is
known for his economic policies, including the North American Free Trade Agreement and th
e Balanced Budget Act of 1997.
Human: Which party was in power back then?
AI:  The Democratic Party was in power during Bill Clinton's presidency in 1996. The Demo
cratic Party is one of the two major political parties in the United States, along with t
he Republican Party. They have historically supported social welfare programs and progres
sive policies.
```

Conclusion

So, there was a brief and quick (I hope lol) intro with Lanchain. Please note, this file merely touches the surface. There are tons of interesting experiments you can do with Langchain, check out their official page: https://python.langchain.com/docs/modules/ (https://python.langchain.com/docs/modules/)

N.B. Langchain is still evolving, they are adding many new functionality, and changing things rapidly. I's strongly encourage to keep an eye on their official page

References

1. https://www.youtube.com/watch?v=2xxziIWmaSA&list=PLqZXAkvF1bPNQER9mLmDbntNfSpzdDIU5&index=3 (https://www.youtube.com/watch?v=2xxziIWmaSA&list=PLqZXAkvF1bPNQER9mLmDbntNfSpzdDIU5&index=3)
2. https://www.youtube.com/watch?v=nAmC7SoVLd8 (https://www.youtube.com/watch?v=nAmC7SoVLd8)
3. https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/ (https://www.deeplearning.ai/short-courses/langchain-for-llm-application-development/)