Evaluation

Contents

Note

Conceptual Guide

This section of documentation covers how we approach and think about evaluation in LangChain. Both evaluation of internal chains/agents, but also how we would recommend people building on top of LangChain approach evaluation.

The Problem

It can be really hard to evaluate LangChain chains and agents. There are two main reasons for this:

# 1: Lack of data

You generally don't have a ton of data to evaluate your chains/agents over before starting a project. This is usually because Large Language Models (the core of most chains/agents) are terrific few-shot and zero shot learners, meaning you are almost always able to get started on a particular task (text-to-SQL, question answering, etc) without a large dataset of examples. This is in stark contrast to traditional machine learning where you had to first collect a bunch of datapoints before even getting started using a model.

Skip to main content

Most chains/agents are performing tasks for which there are not very good metrics to evaluate performance. For example, one of the most common use cases is generating text of some form. Evaluating generated text is much more complicated than evaluating a classification prediction, or a numeric prediction.

The Solution

LangChain attempts to tackle both of those issues. What we have so far are initial passes at solutions - we do not think we have a perfect solution. So we very much welcome feedback, contributions, integrations, and thoughts on this.

Here is what we have for each problem so far:

# 1: Lack of data

We have started LangChainDatasets a Community space on Hugging Face. We intend this to be a collection of open source datasets for evaluating common chains and agents. We have contributed five datasets of our own to start, but we highly intend this to be a community effort. In order to contribute a dataset, you simply need to join the community and then you will be able to upload datasets.

We're also aiming to make it as easy as possible for people to create their own datasets. As a first pass at this, we've added a QAGenerationChain, which given a document comes up with question-answer pairs that can be used to evaluate question-answering tasks over that document down the line. See this notebook for an example of how to use this chain.

# 2: Lack of metrics

We have two solutions to the lack of metrics.

The first solution is to use no metrics, and rather just rely on looking at results by eye to get a sense for how the chain/agent is performing. To assist in this, we have developed (and will continue to develop) tracing, a UI-based visualizer of your chain and agent runs.

The second solution we recommend is to use Language Models themselves to evaluate outputs. For this we have a few different chains and prompts aimed at tackling this issue.

Skip to main content

The Examples

We have created a bunch of examples combining the above two solutions to show how we internally evaluate chains and agents when we are developing. In addition to the examples we've curated, we also highly welcome contributions here. To facilitate that, we've included a template notebook for community members to use to build their own examples.

The existing examples we have are:

Question Answering (State of Union): An notebook showing evaluation of a question-answering task over a State-of-the-Union address.

Question Answering (Paul Graham Essay): An notebook showing evaluation of a question-answering task over a Paul Graham essay.

SQL Question Answering (Chinook): An notebook showing evaluation of a question-answering task over a SQL database (the Chinook database).

Agent Vectorstore: An notebook showing evaluation of an agent doing question answering while routing between two different vector databases.

Agent Search + Calculator: An notebook showing evaluation of an agent doing question answering using a Search engine and a Calculator as tools.

Other Examples

In addition, we also have some more generic resources for evaluation.

Question Answering: An overview of LLMs aimed at evaluating question answering systems in general.

Data Augmented Question Answering: An end-to-end example of evaluating a question answering system focused on a specific document (a RetrievalQAChain to be precise). This example highlights how to use LLMs to come up with question/answer examples to evaluate over, and then highlights how to use LLMs to evaluate performance on those generated

Skip to main content

Hugging Face Datasets: Covers an example of loading and using a dataset from Hugging Face for evaluation.