【prompt 自动优化】DSPy：原理与实践全解析

原创 方方 方方的算法花园 2024年10月31日 09:02 北京

点击蓝字 关注我们

写在前面

在使用prompt方法与大语言模型（LLMs）进行交互的过程中，我们时常会陷入一系列的困扰：

- **当模型的输出未达预期效果时，我们难以判断这究竟是模型自身能力的瓶颈，还是由于我们精心设计的prompt存在缺陷?**
- **当我们尝试切换至其他模型时，之前所使用的prompt往往如同失灵一般，不再能够有效地发挥作用，此时我们便陷入了不知如何调整的困境。**

不同的prompt对同一个模型所产生的效果可能大相径庭，反之，同一prompt应用于不同模型时，其表现也会有显著的区别。那么，在这样的情况下，我们不禁要问，是否存在一种自动化的手段，能够帮助我们生成最为理想的提示，从而摆脱对人工编写和主观直觉的过度依赖呢？毕竟，人工编写prompt的过程充满了不确定性，而在计算机科学这个追求精确性的领域中，这种依赖直觉的传统做法在未来可能会被更为系统、科学的方法所取代。所以，深入研究如何实现prompt技术的自动化优化，已然成为当前至关重要的课题。

下面，便将介绍DSPy，2023年10月斯坦福发布的一个对语言模型Prompt和权重进行算法优化的框架。我们将对其原理和实践进行详细介绍。

DSPy 简介

DSPy：声明式自改进语言程序（Declarative Self-improving Language Programs（in Python）），一个对语言模型Prompt和权重进行算法优化的框架，由斯坦福大学于2023年10月发布。DSPy强调通过编程而非硬编码Prompt构建基于LLM的应用，将构建大模型流水线的过程从操纵基于字符串的prompt技术转变为接近编程的方式。
在没有 DSPy 的情况下使用语言模型构建一个复杂系统，通常需要：
（1）将问题分解为步骤
（2）很好地prompt你的语言模型，直到每个步骤单独运行良好
（3）调整步骤以使其协同工作
（4）生成合成示例来调整每个步骤
（5）使用这些示例微调较小的语言模型以降低成本。
目前，这既困难又混乱：每次你改变pipelines、语言模型或数据时，所有prompt（或微调步骤）可能都需要改变。

为了使这个过程更系统、更稳健，DSPy 做了两件事：**（1）将程序的流程（模块）与每个步骤的参数（语言模型prompt和权重）分开。**
（2）DSPy 引入了新的优化器，这些优化器是由语言模型驱动的算法，可以在给定你想要最大化的指标的情况下，调整语言模型调用的prompt和 / 或权重。

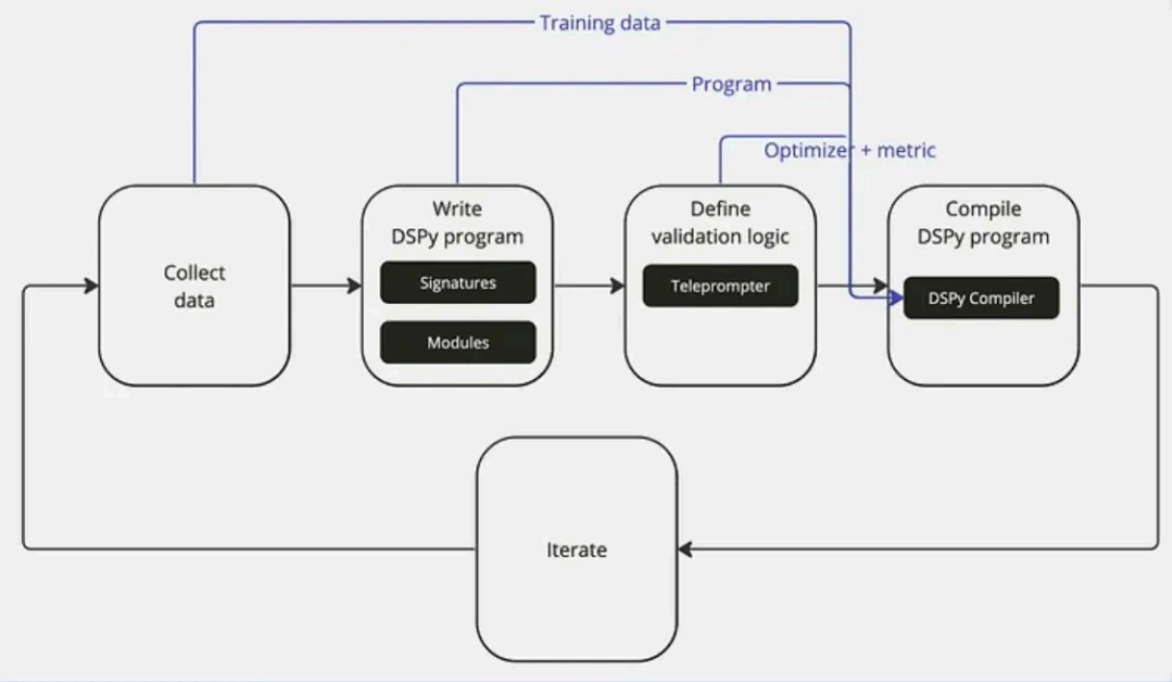
DSPy 可以常规地教导像 GPT-3.5 或 GPT-4 这样强大的模型，以及像 T5-base 或 Llama2-13b 这样的本地模型，使其在任务中更加可靠，即具有更高的质量和 / 或避免特定的失败模式。DSPy 优化器将为每个语言模型将相同的程序“编译”成不同的指令、少量示例prompt

和 / 或权重更新（微调）。这是一种新的范式，在这种范式中，语言模型及其prompt作为一个可以从数据中学习的更大系统的可优化部分逐渐退居幕后。

总的来说：**prompt减少，得分提高，使用语言模型解决困难任务的更系统方法。**

DSPy 工作流程

DSPy 的设计理念源自神经网络抽象共识，其中涵盖了两大关键要点：
◦ **构建一种普适的机制，并通过模块化组合来实现。**
◦ **模型权重可通过优化器来训练，而非手动调整。**

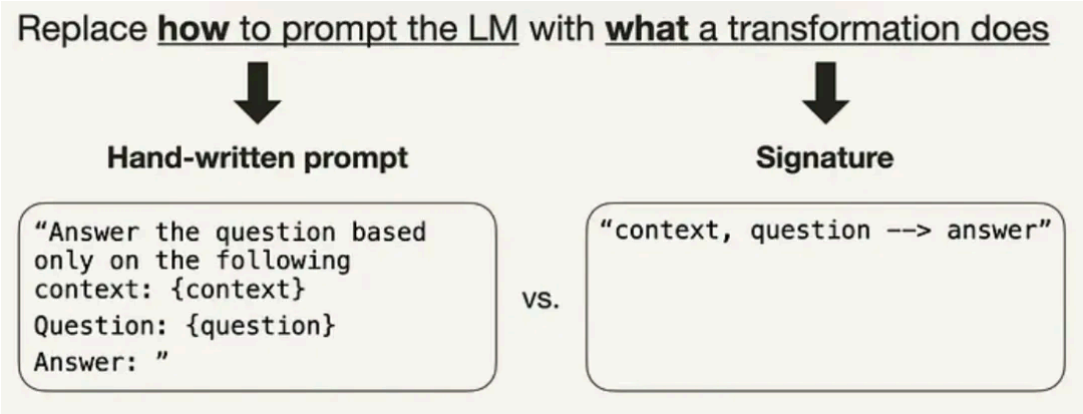


利用 DSPy 构建基于大型语言模型（llm）的应用程序的工作流程如下：
（1）数据收集：收集一些程序输入和输出的示例（比如，问题与答案的对应关系），以便优化你的处理流程。
（2）DSPy 程序编写：使用特征签名（Signature）和模块（Modules）来定义程序的逻辑以及各组件之间的信息传递，以解决你的任务。
（3）验证逻辑定义：确定一种逻辑，利用验证度量和优化器（或提词器 teleprompters）来优化你的程序。
（4）DSPy 程序编译：DSPy 编译器会综合考虑训练数据、程序、优化器和验证度量等因素，来优化你的程序（比如，prompt或fine-tune）。
（5）迭代过程：通过改进数据、程序或验证等方面，不断重复该过程，直至你对处理流程的性能感到满意。

整个过程主要涉及**特征签名（Signatures）、模块（Modules）和提词器（teleprompters，也称为优化器）**这三个组成部分。

> **（1）特征签名（Signatures）：对传统手写prompt和微调的抽象**
> 在 DSPy 程序中，对语言模型的每一次调用都需要具备自然语言特征签名，以替代传

统的手写prompt。特征签名是一个简短的函数，它指明了一项转换的任务内容，而非具体如何提示语言模型去执行（比如，“运用问题和上下文并得出答案”）。



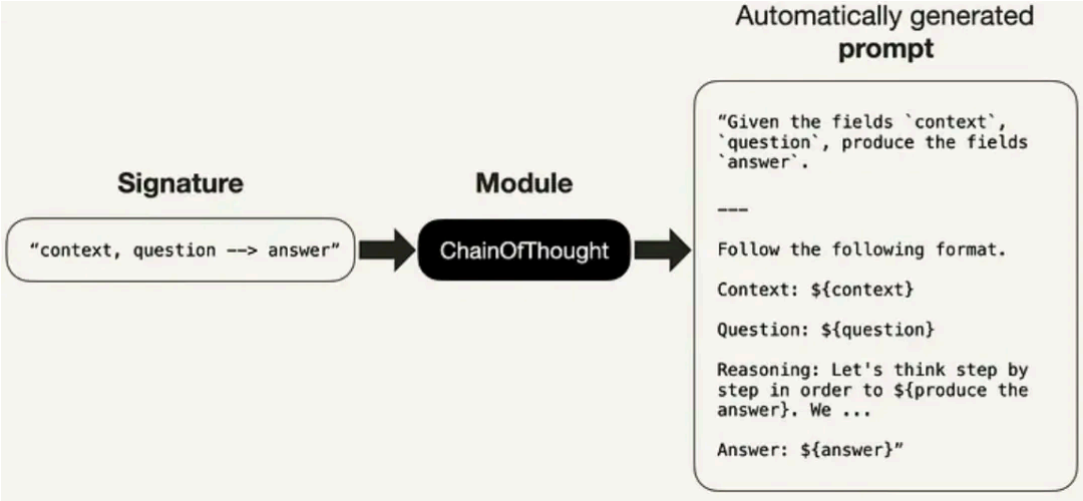
特征签名的简略语法示例通常包含三个要素：语言模型要解决的子任务的最简描述、输入字段的描述、输出字段的描述。

```
"question -> answer"
"long-document -> summary"
"context, question -> answer"
```

相较于手写提示，特征签名可通过为每个特征签名引导示例编译成自我完善和自适应的处理流程的提示或微调。

（2）模块（Modules）：对更高级提示技术的抽象

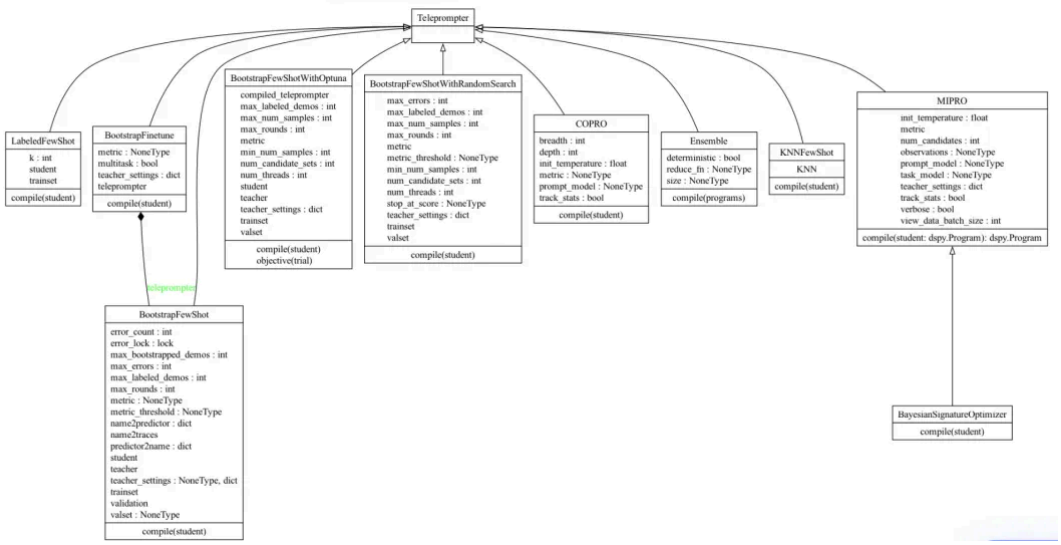
大家可能熟悉多种不同的prompt技巧，比如添加 “Your task is to...” 之类的句子，或在提示开头加上 “You are a...”，或者运用思维链（“让我们一步一步地思考”），亦或是在提示结尾添加 “Don't make anything up” 或 “Only use the provided context” 这样的语句。对此，DSPy 进行了模板化和参数化处理，以抽象这些提示技术，通过应用提示、微调、增强和推理等技术，使 DSPy 特征签名适应任务。



（3）优化器（Optimizers）或提词器（teleprompters）：替代手动提示工程

提词器（Teleprompters）作为 DSPy 程序的优化器，借助特征签名和一个度量或评估系统（可能以语言模型作为评判标准）来开展实验，以确定更理想的提示文本和最佳的少量示例集。其用于改善 DSPy 程序的质量或成本，选择最佳的提示或微调策略。

目前可用的DSPy Optimizers：（详见https://dspy-docs.vercel.app/building-blocks/6-optimizers/）



所有这些都可以通过from dspy.teleprompt import * 访问。

DSPy 的创新之处在于将特征签名、模块和优化器相结合使用，特征签名为语言模型提供指引，优化器利用特征签名和一个度量或评估系统来进行实验，以确定更理想的提示文本和最佳的少量示例集。使用 DSPy 的开发者，只需专注于任务本身，无需纠结于提示工程的具体细节，大大提升了开发效率。

DSPy断言

2023 年 12 月，在论文《DSPy Assertions: Computational Constraints for Self-Refining Language Model Pipelines》中对 LM 断言进行了定义，这是一种用于表达语言模型应遵循的计算约束的编程结构。LM 断言被视为程序元素，它规定了在语言模型流水线执行过程中必须满足的某些条件或规则。这些约束能够确保流水线的行为符合开发者所设定的不变量或准则，进而提升流水线输出的可靠性、可预测性以及正确性。

将 LM 断言融入 DSPy 编程模型中，其作用不仅限于传统的运行时监控。LM 断言还能实现多种新颖的断言驱动优化，从而改进 LM 程序。具体有以下三种方式：

- **断言驱动的回溯：**在推理过程中，LM 断言能够促进语言模型流水线的自我优化。当约束条件不满足时，允许流水线回溯并重新尝试失败的模块。LM 断言会提供关于重试尝试的反馈，它们会将错误输出和错误消息注入提示中，以实现自我优化输出。
- **断言驱动的示例引导：**在编译时，LM 断言可以启用引导提示优化器。与 DSPy 中已有的自动提示优化器相集成，它们可以生成更具挑战性的少量样本示例，从而指导 LM 程序完成具有挑战性的步骤。
- **反例引导：**在提示优化和示例引导过程中，LM 断言的另一个重要贡献是开发包含失败示例和修复错误痕迹的演示。当反例与引导的少量样本示例混合时，语言模型将更有可能避免出现同样的错误，而无需进行断言驱动的回溯。

LM 断言包含两种类型：（硬性）断言和（软性）建议，分别用 Assert 和 Suggest 表示。
硬性断言表示关键条件，当在最大重试次数后仍被违反时，会导致语言模型流水线停止运行，这表明存在不可协商的违反要求的情况。而（软性）建议则表示期望但非必要的属性，其违反会触发自我优化过程，但即使超过最大重试次数，也不会停止流水线。相反，流水线会继续执行下一个模块。

通过将断言整合到 DSPy 中，借助设计和实现的三种新的断言驱动优化，使得 DSPy 程序能够自我优化，并生成符合特定准则的输出。这不仅简化了调试过程，还让开发人员更清楚地了解复杂流水线中的语言模型行为。此外，通过将 LM 断言与 DSPy 中的提示优化器相结合，能够引导生成更好的少量样本示例和反例，使流水线更加稳健和高效。

DSPy 实践

1 环境安装

```
pip install dspy-ai 或
pip install git+https://github.com/stanfordnlp/dspy.git 下载最新版本
```

2 实践案例：入门

（1）Getting Started

首先设置语言模型（LM）和检索模型（RM）

```
turbo = dspy.OpenAI(model='gpt-3.5-turbo')
colbertv2_wiki17_abstracts = dspy.ColBERTv2(url='http://20.102.90.50:2017/wiki
# 默认配置 DSPy 使用 turbo LM 和 ColBERTv2 检索器
dspy.settings.configure(lm=turbo, rm=colbertv2_wiki17_abstracts)
```

（2）Task Examples

从 HotPotQA 多跳数据集中加载一个小样本

```
from dspy.datasets import HotPotQA
从 HotPotQA 多跳数据集中加载一个小样本
# 加载数据集
dataset = HotPotQA(train_seed=1, train_size=20, eval_seed=2023, dev_size=50,
# 告诉DSPy“question”字段是输入。任何其他字段都是标签和/或元数据。
trainset = [x.with_inputs('question') for x in dataset.train]
devset = [x.with_inputs('question') for x in dataset.dev]
len(trainset), len(devset)
# (20, 50)
train_example = trainset[0]
print(f"Question: {train_example.question}")
print(f"Answer: {train_example.answer}")
# Question: At My Window was released by which American singer-songwriter?
# Answer: John Townes Van Zandt
dev_example = devset[18]
print(f"Question: {dev_example.question}")
```

```
print(f"Answer: {dev_example.answer}")
print(f"Relevant Wikipedia Titles: {dev_example.gold_titles}")
# Question: What is the nationality of the chef and restaurateur featured in
# Answer: English
# Relevant Wikipedia Titles: {'Robert Irvine', 'Restaurant: Impossible'}print
```

（3）Building Blocks

以声明式的方式定义模块。

```
# 为基本问答定义一个简单的签名。
class BasicQA(dspy.Signature):
    """Answer questions with short factoid answers."""
    question = dspy.InputField()
    answer = dspy.OutputField(desc="often between 1 and 5 words")
```

```
# 定义预测器。
generate_answer = dspy.Predict(BasicQA)
# 在特定输入上调用预测器。
pred = generate_answer(question=dev_example.question)
# 打印输入和预测结果。
print(f"Question: {dev_example.question}")
print(f"Predicted Answer: {pred.answer}")
# Question: What is the nationality of the chef and restaurateur featured in R
# Predicted Answer: American
```

```
# 为了可见性，我们可以检查这个极其基本的预测器是如何实现我们的签名的。
# 让我们检查一下我们的LM（turbo）的历史。
turbo.inspect_history(n=1)

"""
Answer questions with short factoid answers.
---
Follow the following format.
Question: ${question}
Answer: often between 1 and 5 words
---
Question: What is the nationality of the chef and restaurateur featured in Re
Answer: American
"""
```

```
# 定义预测器。注意，我们只是在更改类。BasicQA 的签名保持不变。
generate_answer_with_chain_of_thought = dspy.ChainOfThought(BasicQA)
# 在相同的输入上调用预测器。
```

```
pred = generate_answer_with_chain_of_thought(question=dev_example.question)
# 打印输入、思维链和预测结果。
print(f"Question: {dev_example.question}")
print(f"Thought: {pred.rationale.split('.', 1)[1].strip()}")
print(f"Predicted Answer: {pred.answer}")
# Question: What is the nationality of the chef and restaurateur featured in
# Thought: We know that the chef and restaurateur featured in Restaurant: Imp
# Predicted Answer: British
```

```
retrieve = dspy.Retrieve(k=3)
topK_passages = retrieve(dev_example.question).passages
print(f"Top {retrieve.k} passages for question: {dev_example.question} \n", '
for idx, passage in enumerate(topK_passages):
     print(f'{idx+1}]', passage, '\n')

"""
Top 3 passages for question: What is the nationality of the chef and restaura
 ------------------------------

1] Restaurant: Impossible | Restaurant: Impossible is an American reality tel

2] Jean Joho | Jean Joho is a French-American chef and restaurateur. He is ch

3] List of Restaurant: Impossible episodes | This is the list of the episodes

"""
```

具体案例可以参考：

（1）入门任务：使用 HotPotQA 处理复杂问题的回答任务（https://github.com/stanfordnlp/dspy/blob/main/intro.ipynb）

（2）教 LMs 推理逻辑陈述和否定（https://github.com/stanfordnlp/dspy/blob/main/examples/nli/scone/scone.ipynb）

（3）使用本地模型和数据集（https://github.com/stanfordnlp/dspy/blob/main/examples/skycamp2023.ipynb）

参考链接

DSPy论文：

- DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines. 将声明式语言模型调用编译为自我改进的pipelines（https://arxiv.org/pdf/2310.03714）

- DSPy Assertions: Computational Constraints for Self-Refining Language Model Pipelines. DSPy 断言：自精炼语言模型pipelines的计算约束。（https://arxiv.org/pdf/2312.13382）

DSPy Github：https://github.com/stanfordnlp/dspy

DSPy 文档：

https://dspy-docs.vercel.app/intro/

其他链接：

- Intro to DSPy: Goodbye Prompting, Hello Programming! (https://towardsdatascience.com/intro-to-dspy-goodbye-prompting-hello-programming-4ca1c6ce3eb9)
- DSPy入门：告别指令提示，拥抱编程之旅！(https://blog.csdn.net/lichunericli/article/details/138136237)
- Prompt或许的新未来， DSPy使用从0到1快速上手（https://mp.weixin.qq.com/s/MYZfhcwshjibQAisgBwalg）

语言模型 5　　#LLM学习 12

语言模型 · 目录

上一篇 · 【prompt 自动优化】TextGrad：借鉴DSPy 并融合梯度下降的新方法