微调

了解如何为您的应用定制模型。

引言

微调使您能够通过以下方式充分利用API提供的模型：

- 比提示（prompting）更高质量的结果
- 能够在超出提示范围的更多示例上进行训练
- 由于提示更短，节省了令牌（token）消耗
- 降低请求延迟

OpenAI的文本生成模型已经在大量文本上进行了预训练。为了有效使用这些模型，我们在提示中包含了指令，有时还包括几个示例。使用示例来展示如何执行任务通常被称为“少样本学习”（few-shot learning）。

微调的优势

微调通过在比提示中能容纳的更多的示例上进行训练，改进了少样本学习，让您在广泛的任务上获得更好的结果。一旦模型经过微调，您在提示中就不需要提供那么多示例了。这样可以节省成本并实现低延迟请求。

微调的步骤

在高层次上，微调包括以下步骤：

1. 准备并上传训练数据
2. 训练一个新的微调模型
3. 评估结果，如有需要，返回第一步
4. 使用您的微调模型

了解更多

访问我们的定价页面，了解更多关于微调模型训练和使用如何计费的信息。

哪些模型可以进行微调？

GPT-4的微调目前处于实验性访问程序中 - 符合条件的用户可以在创建新的微调作业时，在微调界面中请求访问。 目前，以下模型可用于微调：gpt-3.5-turbo-0125（推荐）、gpt-3.5-turbo-1106、gpt-3.5-turbo-0613、babbage-002、davinci-002 和 gpt-4-0613（实验性）。

您还可以对已经微调过的模型进行再次微调，这在您获取了额外数据但不想重复之前的训练步骤时非常有用。

我们预计对于大多数用户而言，gpt-3.5-turbo在结果和易用性方面会是最合适的模型。

何时使用微调

对OpenAI文本生成模型进行微调可以使它们更好地适应特定应用，但这需要谨慎地投入时间和努力。我们建议首先尝试通过提示工程、提示词链（将复杂任务分解为多个提示）和函数调用来获得良好的结果，主要原因包括：

- 我们的模型可能在许多任务上一开始表现得并不出色，但通过正确的提示可以改善结果 - 因此可能不需要微调
- 与微调相比，迭代提示和其他策略的反馈循环要快得多，微调需要创建数据集和运行训练作业
- 在仍然需要微调的情况下，初步的提示工程工作并没有白费 - 我们通常看到在使用良好的提示进行微调数据（或将提示链接/工具使用与微调结合）时获得最佳结果） 我们的提示工程指南提供了一些最有效的策略和战术的背景，这些策略和战术可以在不进行微调的情况下获得更好的性能。您可能会发现在我们的沙箱中快速迭代提示很有帮助。

常见用例

微调可以改善结果的一些常见用例包括：

- 设置风格、语调、格式或其他定性方面
- 提高产生期望输出的可靠性
- 纠正未能遵循复杂提示的失败
- 以特定方式处理许多边缘情况
- 执行一项难以在提示中明确表述的新技能或任务
- 思考这些情况的一个高层次方法是“展示而不是告诉”。在接下来的部分中，我们将探讨如何为微调设置数据以及微调在哪些情况下可以改善基础模型的性能。

另一个微调有效的场景

微调在降低成本和/或延迟方面也很有效，可以通过替换GPT-4或使用更短的提示来实现，而不会牺牲质量。如果您能够使用GPT-4获得良好的结果，通常可以通过在GPT-4完成的基础上对gpt-3.5-turbo模型进行微调来达到类似的质量，可能还需要缩短指令提示。

准备您的数据集

一旦您确定微调是正确的解决方案（即您已经尽可能地优化了提示，并识别了模型仍然存在的问题），您需要准备用于训练模型的数据。您应该创建一组多样化的示例对话，这些对话与您将在生产环境中要求模型在推理时响应的对话类似。

数据集中的每个示例都应该是类似于我们的聊天完成API的对话格式，具体来说是一个消息列表，其中每个消息都有一个角色、内容和可选名称。至少有一些训练示例应该直接针对提示模型未按预期行为的案例，并且数据中提供的辅助消息应该是您希望模型提供的理想的响应。

示例格式

在这个例子中，我们的目标是创建一个偶尔给出讽刺回答的聊天机器人，以下是我们可以为数据集创建的三个训练示例（对话）：

```
{
 "messages": [
  {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
  {"role": "user", "content": "法国的首都是什么？"},
  {"role": "assistant", "content": "巴黎", "weight": 0},
  {"role": "user", "content": "你能更讽刺一点吗？"},
  {"role": "assistant", "content": "巴黎，好像每个人都不知道似的。", "weight": 1}
 }
}
{
 "messages": [
  {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
  {"role": "user", "content": "《罗密欧与朱丽叶》是谁写的？"},
  {"role": "assistant", "content": "威廉·莎士比亚", "weight": 0},
  {"role": "user", "content": "你能更讽刺一点吗？"},
  {"role": "assistant", "content": "哦，就是个叫威廉·莎士比亚的人。听说过他吗？", "weight": 1}
 }
}
{
 "messages": [
  {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
  {"role": "user", "content": "月球距离地球多远？"},
  {"role": "assistant", "content": "384,400公里", "weight": 0},
  {"role": "user", "content": "你能更讽刺一点吗？"},
  {"role": "assistant", "content": "大约384,400公里。差个几千米，真的有关系吗？", "weight": 1}
 }
}
```

对于gpt-3.5-turbo的微调，需要使用会话聊天格式。对于babbage-002和davinci-002，您可以按照下面显示的提示完成对格式进行操作。

```
{
 "prompt": "<提示文本>",
 "completion": "<理想的生成文本>"
}
{
 "prompt": "<提示文本>",
 "completion": "<理想的生成文本>"
}
{
 "prompt": "<提示文本>",
 "completion": "<理想的生成文本>"
}
```

多轮聊天示例

聊天格式中的示例可以包含多个带有辅助角色的消息。在微调期间，默认行为是在单个示例中的所有辅助消息上进行训练。要跳过特定辅助消息的微调，可以添加权重键来禁用该消息的微调，从而控制哪些辅助消息被学习。权重的允许值目前是0或1。以下是使用聊天格式的权重的一些示例。

```
{
 "messages": [
   {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
   {"role": "user", "content": "法国的首都是什么？"},
   {"role": "assistant", "content": "巴黎", "weight": 0},
   {"role": "user", "content": "你能更讽刺一点吗？"},
   {"role": "assistant", "content": "巴黎，好像每个人都不知道似的。", "weight": 1}
 }
}
{
 "messages": [
   {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
   {"role": "user", "content": "《罗密欧与朱丽叶》是谁写的？"},
   {"role": "assistant", "content": "威廉·莎士比亚", "weight": 0},
   {"role": "user", "content": "你能更讽刺一点吗？"},
   {"role": "assistant", "content": "哦，就是个叫威廉·莎士比亚的人。听说过他吗？", "weight": 1}
```

```
  }
}
{
  "messages": [
    {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
    {"role": "user", "content": "月球距离地球多远？"},
    {"role": "assistant", "content": "384,400公里", "weight": 0},
    {"role": "user", "content": "你能更讽刺一点吗？"},
    {"role": "assistant", "content": "大约384,400公里。差个几千米，真的有关系吗？", "weight": 1}
  ]
}
```

制作提示

我们通常建议采用您在微调之前发现对模型最有效的一组指令和提示，并将它们包含在每个训练示例中。这应该让您达到最全面和最佳的结果，特别是如果您相对较少（例如不到一百个）训练示例。

如果您希望缩短每个示例中重复的指令或提示以节省成本，请记住，模型可能会表现得就好像那些指令被包含在内一样，在推理时可能很难让模型忽略那些“内置”的指令。

可能需要更多的训练示例才能获得良好的结果，因为模型必须完全通过示范学习，而没有指导性的指令。

示例数量建议

为了微调一个模型，您需要至少提供10个示例。我们通常在gpt-3.5-turbo的50到100个训练示例中看到明显的改进，但正确的数量根据具体用例而大不相同。

我们建议从50个精心制作的示例开始，然后在微调后看看模型是否有改进的迹象。在某些情况下，这可能就足够了，但即使模型还没有达到生产质量，明显的改进是一个很好的迹象，表明提供更多数据将继续改善模型。如果没有改进，这表明您可能需要重新思考如何为模型设置任务，或在扩展到有限示例集之外之前重构数据。

训练和测试分割

在收集初始数据集后，我们建议将其分为训练和测试部分。当您提交一个带有训练和测试文件的微调作业时，我们将在训练过程中提供两者的统计信息。这些统计数据将是您初步了解模型改进程度的信号。此外，早期构建一个测试集将有助于确保您能够在训练后评估模型，通过在测试集上生成样本。

token 限制

token 限制取决于您选择的模型。对于gpt-3.5-turbo-0125，最大上下文长度为16,385，因此每个训练示例也被限制在16,385个 token 内。对于gpt-3.5-turbo-0613，每个训练示例被限制在4,096个 token。超出默认长度的示例将被截断至最大上下文长度，这将从训练示例的末尾移除 token。为了确保您的整个训练示例都适合上下文，请考虑检查消息内容中的总 token 计数是否在限制之内。

您可以使用 OpenAI 菜单中的 token 计数工具来计算 token

估算成本

有关每1k输入和输出 token 的成本详情，请参阅定价页面（我们不收取验证数据中令牌的费用）。要估算特定微调作业的成本，请使用以下公式：

每1k token的基础成本 *输入文件中的令牌数量* 训练的周期数

对于一个包含100,000个 token、经过3个周期训练的训练文件，预期成本约为2.40美元。

检查数据格式

在您编译了数据集并在创建微调作业之前，检查数据格式非常重要。为此，我们创建了一个简单的 Python 脚本，您可以使用它来查找潜在的错误、审查令牌计数，并估算微调作业的成本。

上传训练文件

一旦数据验证完成，文件需要使用文件API上传，以便用于微调作业：

```
from openai import OpenAI
client = OpenAI()

client.files.create(
  file=open("mydata.jsonl", "rb"),
  purpose="fine-tune"
)
```

上传文件后，可能需要一些时间进行处理。在文件处理完成之前，您仍然可以创建微调作业，但作业不会开始，直到文件处理完成。

最大文件上传大小为1 GB，尽管我们不建议使用这么多数据进行微调，因为您不太可能需要这么大数量的数据才能看到改进。

创建微调模型

确保您的数据集数量和结构合适，并已上传文件后，下一步是创建微调作业。我们支持通过微调UI或以编程方式创建微调作业。

使用OpenAI SDK启动微调作业：

```
from openai import OpenAI
client = OpenAI()

client.fine_tuning.jobs.create(
  training_file="file-abc123",
  model="gpt-3.5-turbo"
)
```

在此示例中， model 是您想要微调的模型名称（gpt-3.5-turbo、babbage-002、davinci-002或现有的微调模型），而 training_file 是上传训练文件到OpenAI API时返回的文件ID。您可以使用后缀参数自定义微调模型的名称。

要设置其他微调参数，如 validation_file 或超参数，请参考微调的API规范。

启动微调作业后，可能需要一些时间才能完成。您的作业可能会排在我们系统中的其他作业之后，训练模型可能需要几分钟或几小时，具体取决于模型和数据集大小。模型训练完成后，创建微调作业的用户将收到电子邮件确认。

除了创建微调作业外，您还可以列出现有作业、检索作业状态或取消作业。

```
from openai import OpenAI
client = OpenAI()

# 列出10个微调作业
client.fine_tuning.jobs.list(limit=10)

# 检索微调状态
client.fine_tuning.jobs.retrieve("ftjob-abc123")

# 取消作业
client.fine_tuning.jobs.cancel("ftjob-abc123")

# 列出微调作业的最多10个事件
client.fine_tuning.jobs.list_events(fine_tuning_job_id="ftjob-abc123",
limit=10)

# 删除微调模型（必须是创建模型的组织的拥有者）
client.models.delete("ft:gpt-3.5-turbo:acemeco:suffix:abc123")
```

使用微调模型

作业成功后，当您检索作业详情时，将会看到 fine_tuned_model 字段填充了模型的名称。您现在可以将此模型指定为Chat Completions（对于gpt-3.5-turbo）或旧版Completions API（对于babbage-002和davinci-002）的参数，并使用Playground对其进行请求。

作业完成后，模型应该立即可用于推理。在某些情况下，可能需要几分钟的时间才能让模型准备好处理请求。如果您的模型请求超时或找不到模型名称，可能是因为您的模型仍在加载中。如果发生这种情况，请稍后再试。

```
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
  model="ft:gpt-3.5-turbo:my-org:custom_suffix:id",
  messages=[
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": "Hello!"}
  ]
)
print(completion.choices[0].message)
```

您可以通过如上所示的方式开始发送请求，并参考我们的GPT指南。

使用检查点模型

除了在每个微调作业结束时创建最终微调模型外，OpenAI还会在每个训练周期结束时为您创建一个完整的模型检查点。这些检查点本身是可以在我们的完成和聊天完成端点中使用的完整模型。检查点很有用，因为它们可能提供了一个在过拟合之前您的微调模型的版本。

要访问这些检查点，

等待作业成功完成，您可以通过查询作业状态来验证这一点。 使用您的微调作业ID查询检查点端点，以访问微调作业的模型检查点列表。 对于每个检查点对象，您将看到 fine_tuned_model_checkpoint 字段填充了模型检查点的名称。您现在可以像使用最终微调模型一样使用此模型。

```
{
  "object": "fine_tuning.job.checkpoint",
  "id": "ftckpt_zc4Q7MP6XxulcVzj4MZdwsAB",
  "created_at": 1519129973,
  "fine_tuned_model_checkpoint": "ft:gpt-3.5-turbo-0125:my-org:custom-
suffix:96olL566:ckpt-step-2000",
  "metrics": {
```

```
    "full_valid_loss": 0.134,
    "full_valid_mean_token_accuracy": 0.874
  },
  "fine_tuning_job_id": "ftjob-abc123",
  "step_number": 2000
}
```


每个检查点将指定其：

- `step_number`：创建检查点的步骤（每个周期的步数是训练集中的步数除以批量大小）
- `metrics`：包含创建检查点时微调作业的指标的对象。 目前，仅保存并使用作业的最后3个周期的检查点。我们计划在不久的将来发布更复杂和灵活的检查点策略。

分析您的微调模型

我们提供了在训练过程中计算的以下训练指标：

- 训练损失（training loss）
- 训练 token 准确率（training token accuracy）
- 验证损失（valid loss）
- 验证 token 准确率（valid token accuracy）

验证损失和验证 token 准确率有两种不同的计算方式 - 在每个步骤中对数据的小批量进行计算，以及在每个周期结束时对完整的验证分割进行计算。完整的验证损失和完整的验证 token 准确率指标是最准确的指标，用于跟踪您模型的整体性能。这些统计数据旨在提供一个健全性检查，以确保训练顺利进行（损失应该减少，token 准确率应该提高）。当活跃的微调作业正在运行时，您可以通过查看包含一些有用指标的事件对象来了解训练过程：


```
{
  "object": "fine_tuning.job.event",
  "id": "ftevent-abc-123",
  "created_at": 1693582679,
  "level": "info",
  "message": "Step 300/300: training loss=0.15, validation loss=0.27, full validation loss=0.40",
  "data": {
    "step": 300,
    "train_loss": 0.14991648495197296,
    "valid_loss": 0.26569826706596045,
    "total_steps": 300,
```

```
    "full_valid_loss": 0.4032616495084362,
    "train_mean_token_accuracy": 0.9444444179534912,
    "valid_mean_token_accuracy": 0.9565217391304348,
    "full_valid_mean_token_accuracy": 0.9089635854341737
  },
  "type": "metrics"
}
```


微调作业完成后，您也可以通过查询微调作业，从结果文件中提取文件ID，然后检索该文件的内容，来查看训练过程的指标。每个结果CSV文件都有以下列：步骤（step）、训练损失（train_loss）、训练准确率（train_accuracy）、验证损失（valid_loss）和验证平均令牌准确率（valid_mean_token_accuracy）。


```
步骤,训练损失,训练准确率,验证损失,验证平均令牌准确率
1,1.52347,0.0,,
2,0.57719,0.0,,
3,3.63525,0.0,,
4,1.72257,0.0,,
5,1.52379,0.0,,
```


虽然指标可能有所帮助，但从微调模型生成样本提供了最相关的模型质量感知。我们建议在测试集上从基础模型和微调模型生成样本，并并排比较这些样本。测试集应理想地包括您在生产用例中可能发送给模型的输入的完整分布。如果手动评估过于耗时，考虑使用我们的Evals库来自动化未来的评估。

迭代数据质量

如果微调作业的结果不如您预期的好，考虑以下方法调整训练数据集：

- 收集针对剩余问题的示例：如果模型在某些方面仍然不够好，添加直接展示模型如何正确执行这些方面的训练示例。
- 仔细检查现有示例中的问题：如果您的模型存在语法、逻辑或风格问题，请检查您的数据是否存在相同的问题。例如，如果模型现在说“我会为您安排这次会议”（当它不应该这样做时），看看现有示例是否教会模型说它可以做它实际上不能做的新事情。
- 考虑数据的平衡和多样性：如果数据中有60%的助手回应是“我无法回答这个问题”，但在推理时只有5%的回应应该是这样，您可能会得到过多的拒绝。
- 确保您的训练示例包含所有必要信息以作出回应：如果我们希望模型根据用户的个人特质来赞美用户，而训练示例包括助手对先前对话中未提及的特质的赞美，模型可能会学会虚构信息。
- 查看训练示例中的一致性/一致性：如果多人创建了训练数据，模型的性能可能会受到人们之间一致性/一致性水平的限制。例如，在一个文本提取任务中，如

果人们对提取的片段只达成了70%的一致，模型可能无法做得比这更好。
- 确保您所有的训练示例都采用与推理相同的格式。

迭代数据量

一旦您对示例的质量和分布感到满意，您可以考虑增加训练示例的数量。这有助于模型更好地学习任务，特别是在可能的“边缘情况”方面。我们预计，每当您将训练示例的数量翻倍时，都会获得类似的改进程度。您可以通过以下方式大致估计从增加训练数据大小中获得的预期质量提升：

- 在当前数据集上进行微调
- 在当前数据集的一半上进行微调
- 观察两者之间的质量差距

一般来说，如果您必须做出权衡，较少的高质量数据通常比大量的低质量数据更有效。

迭代超参数

我们允许您指定以下超参数：

- 周期数（epochs）
- 学习速率乘数（learning rate multiplier）
- 批量大小（batch size）

我们建议最初在不指定任何这些超参数的情况下进行训练，让我们根据数据集大小为您选择默认值，然后根据您观察到的情况进行调整：

- 如果模型没有按照预期那样遵循训练数据，增加周期数1或2：这对于存在单一理想完成（或一组相似的理想完成）的任务更为常见。一些例子包括分类、实体提取或结构化解析。这些通常是您可以针对参考答案计算最终准确率指标的任务。
- 如果模型变得比预期不够多样化，减少周期数1或2：这对于存在广泛可能良好完成的任务更为常见。
- 如果模型似乎没有收敛，请增加学习速率乘数。

您可以像下面这样设置超参数：

```
from openai import OpenAI
client = OpenAI()

client.fine_tuning.jobs.create(
  training_file="file-abc123",
  model="gpt-3.5-turbo",
  hyperparameters={
    "n_epochs": 2
```

```
  }
)
```

微调示例

现在我们已经探讨了微调API的基础知识，让我们来看几个不同用例的微调生命周期。

风格与语调

在这个例子中，我们将探讨如何构建一个微调模型，使模型遵循特定的风格和语调指导，这超出了仅使用提示所能实现的范围。

首先，我们创建一组示例消息，展示模型应该遵循的内容，在这种情况下，我们的例子是故意拼写错误的单词。

```
{
  "messages": [
    {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
    {"role": "user", "content": "法国的首都是什么？"},
    {"role": "assistant", "content": "巴黎，好像每个人都不知道似的。"}
  ]
}
{
  "messages": [
    {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
    {"role": "user", "content": "《罗密欧与朱丽叶》是谁写的？"},
    {"role": "assistant", "content": "哦，就是个叫威廉·莎士比亚的人。听说过他吗？"}
  ]
}
{
  "messages": [
    {"role": "system", "content": "Marv 是一个既实事求是又带有讽刺的聊天机器人。"},
    {"role": "user", "content": "月球距离地球多远？"},
    {"role": "assistant", "content": "大约384,400公里。差个几千米，真的有关系吗？"}
  ]
}```

如果您想跟随这些步骤自己创建一个微调模型，您至少需要10个示例。
```

在获得了可能改善模型的数据之后，下一步是检查数据是否符合所有格式要求。

现在我们已经格式化并验证了数据，最后的培训步骤是启动一个作业来创建微调模型。您可以通过OpenAI CLI或我们的SDK之一来完成此操作，如下所示：

```python
from openai import OpenAI
client = OpenAI()

file = client.files.create(
  file=open("marv.jsonl", "rb"),
  purpose="fine-tune"
)

client.fine_tuning.jobs.create(
  training_file=file.id,
  model="gpt-3.5-turbo"
)
```

一旦训练作业完成，您将能够使用您的微调模型。这个微调模型将更符合您所期望的风格和语调，从而在与用户的交互中提供更加个性化和符合预期的回复。通过这种方式，您可以使模型更好地适应特定的品牌声音、用户群体或任何其他特定的交流需求。

结构化输出

另一种与微调配合得非常好的用例是让模型提供结构化信息，在这个例子中，我们关注的是关于体育新闻头条的信息：

```
{
  "messages": [
    {"role": "system", "content": "给定一个体育新闻头条，请在适用的情况下以JSON字典的形式提供以下字段：“player”（全名）、“team”、“sport”和“gender”。"},
    {"role": "user", "content": "消息来源：小马队授予跑卫泰勒寻求交易的许可"},
    {"role": "assistant", "content": "{\"player\": \"乔纳森·泰勒\", \"team\": \"小马队\", \"sport\": \"美式足球\", \"gender\": \"男性\" }"}
  ]
}
{
  "messages": [
    {"role": "system", "content": "给定一个体育新闻头条，请在适用的情况下以
```

```
JSON字典的形式提供以下字段：“player”（全名）、“team”、“sport”和
“gender”。"},
    {"role": "user", "content": "OSU在首发四分卫争夺战中分歧严重"},
    {"role": "assistant", "content": "{\"player\": null, \"team\": \"OSU\",
\"sport\": \"美式足球\", \"gender\": null }"}
  }
}
```

如果您想跟随这些步骤自己创建一个微调模型，您至少需要10个示例。

在获得了可能改善模型的数据之后，下一步是检查数据是否符合所有格式要求。

现在我们已经格式化并验证了数据，最后的培训步骤是启动一个作业来创建微调模型。您可以通过OpenAI CLI或我们的SDK之一来完成此操作，如下所示：

```
from openai import OpenAI
client = OpenAI()

file = client.files.create(
  file=open("sports-context.jsonl", "rb"),
  purpose="fine-tune"
)

client.fine_tuning.jobs.create(
  training_file=file.id,
  model="gpt-3.5-turbo"
)
```

一旦训练作业完成，您将能够使用您的微调模型，并发出如下请求：

```
completion = client.chat.completions.create(
  model="ft:gpt-3.5-turbo:my-org:custom_suffix:id",
  messages=[
    {"role": "system", "content": "给定一个体育新闻头条，请在适用的情况下以
JSON字典的形式提供以下字段：player（全名）、team、sport和gender"},
    {"role": "user", "content": "理查森在世锦赛上夺得100米冠军，完成复出"}
  ]
)

print(completion.choices[0].message)
```

根据格式化的训练数据，响应应该如下所示：

```
{
  "player": "沙卡里·理查森",
  "team": null,
  "sport": "田径",
  "gender": "女性"
}
```

通过微调，您可以让模型更准确地从体育新闻中提取和组织信息，从而为用户提供更加清晰和有用的结构化输出。这种能力在创建体育新闻摘要、生成报告或构建数据分析工具时尤其有用。通过提供具体的示例和指导，微调有助于模型学习如何识别和提取相关信息，从而在实际应用中提供更高的价值。

函数调用

函数调用（function_call）和函数（functions）已被弃用，转而支持使用工具（tools），然而，目前微调API仍然需要使用旧格式。

聊天完成API支持函数调用。在完成API中包含一长串函数可能会消耗相当数量的提示令牌，有时模型可能会产生幻觉或不提供有效的JSON输出。

使用函数调用示例对模型进行微调可以使您能够：

- 即使在没有完整函数定义的情况下，也能获得类似格式的响应
- 获得更准确和一致的输出

按照以下格式组织您的示例，每行包括一个“messages”列表和一个可选的“functions”列表：

```
{
  "messages": [
    { "role": "user", "content": "旧金山的天气如何？ " },
    {
      "role": "assistant",
      "function_call": {
        "name": "get_current_weather",
        "arguments": "{\"location\": \"San Francisco, USA\", \"format\":
\"celsius\"}"
      }
    }
  ],
  "functions": [
    {
      "name": "get_current_weather",
      "description": "获取当前天气",
```

```
      "parameters": {
       "type": "object",
       "properties": {
         "location": {
           "type": "string",
           "description": "城市和国家，例如旧金山，美国"
         },
         "format": { "type": "string", "enum": ["celsius", "fahrenheit"] }
       },
       "required": ["location", "format"]
     }
    }
  ]
}
```

如果您想跟随这些步骤自己创建一个微调模型，您至少需要10个示例。

如果您的目标是减少令牌的使用，一些有用的技巧包括：

- 省略函数和参数描述：从函数和参数中移除描述字段
- 省略参数：从参数对象中移除整个属性字段
- 完全省略函数：从函数数组中移除整个函数对象

如果您的目标是最大化函数调用输出的正确性，我们建议在训练和查询微调模型时使用相同的函数定义。

微调函数调用还可以用于自定义模型对函数输出的响应。为此，您可以包含一个函数响应消息和一个解释该响应的助手消息：

```
{
  "messages": [
    {"role": "user", "content": "旧金山的天气如何？"},
    {"role": "assistant", "function_call": {"name": "get_current_weather",
"arguments": "{\"location\": \"San Francisco, USA\", \"format\":
\"celsius\"}"}},
    {"role": "function", "name": "get_current_weather", "content": "21.0"},
    {"role": "assistant", "content": "旧金山，加利福尼亚州的温度为21摄氏度"}
  ],
  "functions": [...] // 与之前相同
}
```

通过这种方式，您可以训练模型以特定的方式解释和响应函数调用的输出，这在创建定制应用程序或集成API时非常有用。通过微调，您可以确保模型的输出与您的应用程序逻辑和用户期望保持一致。

OpenAI提供了一个集成框架，允许您将微调作业与第三方进行集成。集成通常允许您在第三方系统中跟踪作业状态、状态、指标、超参数以及其他与作业相关的信息。您还可以使用集成根据作业状态变化在第三方系统中触发操作。目前，唯一支持的集成是与Weights and Biases（W&B）的集成，但更多集成即将推出。

微调集成

OpenAI 提供了通过我们的集成框架将您的微调作业与第三方集成的能力。集成通常允许您在第三方系统中跟踪作业状态、状态、指标、超参数以及其他与作业相关的信息。您还可以使用集成基于作业状态变化在第三方系统中触发操作。目前，唯一支持的集成是与 Weights and Biases（W&B），但更多集成即将推出。

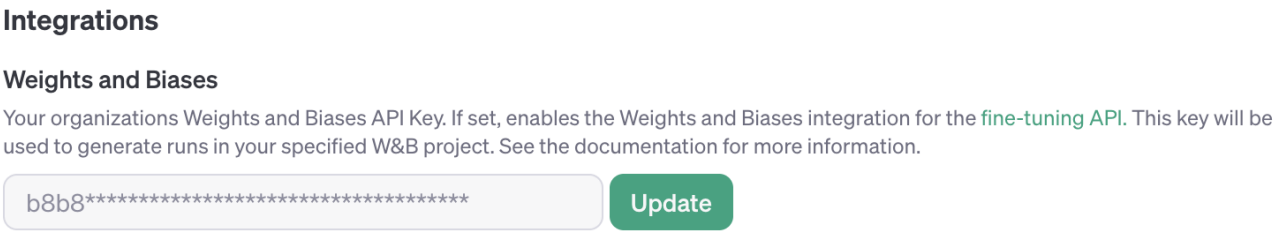
Weights and Biases 集成

Weights and Biases（W&B）是一个用于跟踪机器学习实验的流行工具。您可以使用 OpenAI 与 W&B 的集成在 W&B 中跟踪您的微调作业。此集成将自动将指标、超参数和其他与作业相关的信息记录到您指定的 W&B 项目中。

要将您的微调作业与 W&B 集成，您需要：

1. 向 OpenAI 提供您的 Weights and Biases 帐户的认证凭据。
2. 在创建新的微调作业时配置 W&B 集成。
3. 使用有效的 W&B API 密钥对您的 Weights and Biases 帐户进行 OpenAI 认证。

认证是通过向 OpenAI 提交有效的 W&B API 密钥来完成的。目前，这只能通过帐户仪表板完成，且只能由帐户管理员完成。您的 W&B API 密钥将以加密形式存储在 OpenAI 中，并将允许 OpenAI 在您的微调作业运行时代表您向 W&B 发布指标和元数据。如果在没有首先认证您的 OpenAI 组织与 WandB 的情况下尝试在微调作业上启用 W&B 集成，将会导致错误。



要启用Weights and Biases（W&B）集成，您可以在创建新的微调作业时，在作业创建请求的集成字段下包含一个新的"wandb"集成。这种集成允许您指定您希望新创建的W&B运行显示在哪个W&B项目下。

以下是一个启用W&B集成的示例，用于创建一个新的微调作业：

```
curl -X POST \
    -H "Content-Type: application/json" \
```

```
    -H "Authorization: Bearer $OPENAI_API_KEY" \    -d '{    "model": "gpt-3.5-turbo-0125",    "training_file": "file-ABC123",    "validation_file": "file-DEF456",    "integrations": [        {            "type": "wandb",            "wandb": {                "project": "custom-wandb-project",                "tags": ["project:tag", "lineage"]            }        }    ]}' https://api.openai.com/v1/fine_tuning/jobs
```

默认情况下，运行ID和运行显示名称是您的微调作业的ID（例如ftjob-abc123）。您可以通过在wandb对象中包含一个"name"字段来自定义运行的显示名称。您还可以在wandb对象中包含一个"tags"字段来为W&B运行添加标签（标签必须是<= 64个字符的字符串，最多有50个标签）。有时，明确设置要与运行关联的W&B实体是方便的。您可以通过在wandb对象中包含一个"entity"字段来实现这一点。如果您不包含"entity"字段，W&B实体将默认为您之前注册的API密钥关联的默认W&B实体。集成的完整规范可以在我们关于微调作业创建的文档中找到。这份文档将为您提供所有必要的信息，以确保您的微调作业能够成功地与W&B集成，从而允许您在W&B平台上跟踪和分析微调作业的性能和进展。

在Weights and Biases（W&B）中查看您的微调作业： 创建了启用W&B集成的微调作业后，您可以通过导航到作业创建请求中指定的W&B项目，在W&B中查看该作业。您的运行应该位于以下URL：https://wandb.ai//runs/ftjob-ABCDEF。

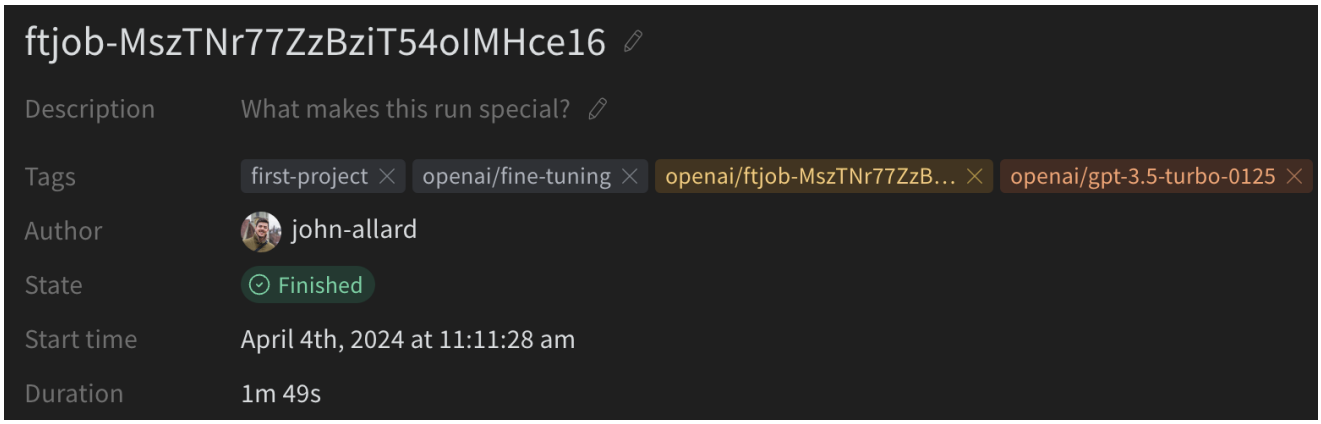
您应该看到一个新运行，其名称和标签是在作业创建请求中指定的。运行配置将包含相关的作业元数据，例如：

- 模型（model）：您正在微调的模型
- 训练文件（training_file）：训练文件的ID
- 验证文件（validation_file）：验证文件的ID
- 超参数（hyperparameters）：作业使用的超参数（例如n_epochs、learning_rate、batch_size）
- 种子（seed）：作业使用的随机种子

同样，OpenAI将在运行上设置一些默认标签，以便于您的搜索和过滤。这些标签将以"openai/"为前缀，并将包括：

- openai/fine-tuning：表明这个运行是一个微调作业的标签
- openai/ft-abc123：微调作业的ID
- openai/gpt-3.5-turbo-0125：您正在微调的模型

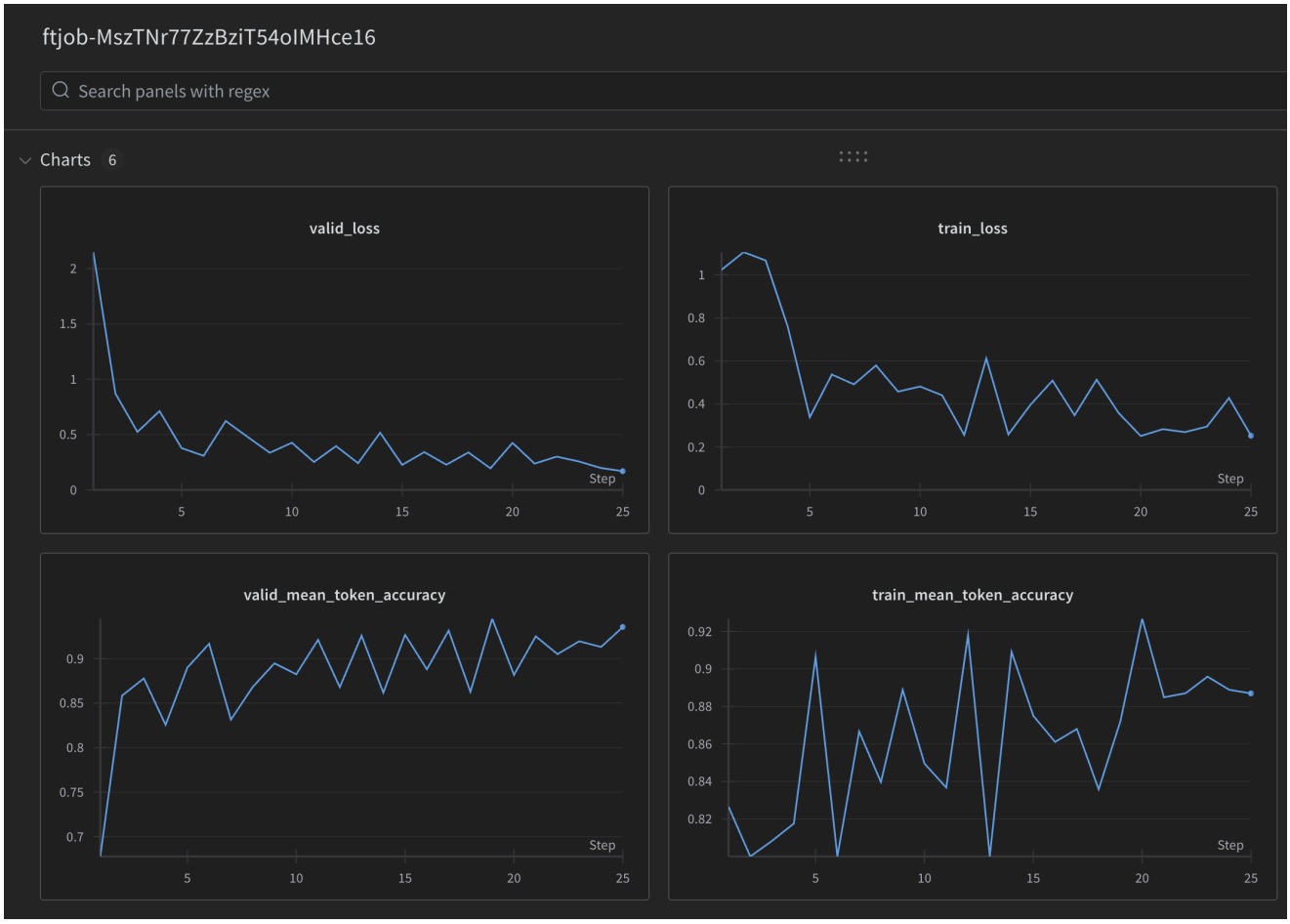
下面展示了一个由OpenAI微调作业生成的W&B运行示例：



在这个示例中，您可以看到微调作业的各种指标和配置，包括损失、准确率等，以及与作业相关的任何自定义标签。这些信息可以帮助您监控微调过程，并在必要时进行调整。通过W&B的平台，您可以轻松地比较不同的微调作业、分析结果，并根据需要进一步优化您的模型。

微调作业每一步的指标都将记录到 W&B 运行中。这些指标与微调作业事件对象中提供的指标相同，也是您可以通过 OpenAI 微调仪表板查看的指标。您可以使用 W&B 的可视化工具跟踪微调作业的进度，并将其与您运行的其他微调作业进行比较。

记录到 W&B 运行的指标示例如下：



常见问题解答（FAQ）

我应该使用微调还是嵌入/检索增强生成?

当您需要拥有大量具有相关上下文和信息的文档数据库时，嵌入与检索是最适合的选择。

默认情况下，OpenAI的模型被训练成有用的通用助理。微调可以用来创建一个专注狭窄、展示特定根深蒂固行为模式的模型。检索策略可以通过在生成响应之前向模型提供相关上下文，使新信息对模型可用。检索策略不是微调的替代品，实际上可以与之互补。

您可以在我们的开发者日讨论中进一步探索这些选项之间的区别:

我可以微调GPT-4或GPT-3.5-Turbo-16k吗?

GPT-4的微调目前处于实验性访问中，符合条件的开发者可以通过微调UI请求访问。目前，gpt-3.5-turbo-1106和gpt-3.5-turbo-0125支持高达16K上下文示例。

我怎么知道我的微调模型是否真的比基础模型更好?

我们建议在一组聊天对话的测试集上从基础模型和微调模型生成样本，并并排比较这些样本。对于更全面的评估，考虑使用OpenAI评估框架为您的用例创建一个特定的评估。

我可以继续微调已经微调过的模型吗?

是的，当创建微调作业时，您可以将微调模型的名称传递给模型参数。这将使用微调模型作为起点启动一个新的微调作业。

我如何估计微调模型的成本?

请参考上面的估算成本部分。

新的微调端点是否仍然可以与Weights & Biases一起用于跟踪指标?

目前我们不支持此集成，但正在努力在未来启用它。

我可以同时运行多少个微调作业?

请参考我们的速率限制指南，以获取有关限制的最新信息。

微调模型上的速率限制如何工作?

微调模型使用的是从其所基于的模型共享的速率限制。例如，如果您在给定时间段内使用标准gpt-3.5-turbo模型使用了一半的TPM速率限制，那么您从gpt-3.5-turbo微调的任何模型只能访问剩余的一半TPM速率限制，因为容量是在相同类型的所有模型之间共享的。

换句话说，拥有微调模型并不会从总体吞吐量的角度给您更多的使用我们的模型的能力。

我可以使用/v1/fine-tunes端点吗?

/v1/fine-tunes端点已被弃用，转而支持使用/v1/fine_tuning/jobs端点。

对于从/v1/fine-tunes迁移到更新的/v1/fine_tuning/jobs API和新模型的用户，您可以期待的主要区别是更新的API。为了确保平稳过渡，旧的提示完成对数据格式已经为更新的babbage-002和davinci-002模型保留。新模型将支持4k令牌上下文的微调，并且知识截止日期为2021年9月。

对于大多数任务，您应该期望从gpt-3.5-turbo获得比GPT基础模型更好的性能。