gguf

This is a Python package for writing binary files in the GGUF (GGML Universal File) format.

- 这是一个用于以GGUF(GGML 通用文件) 格式编写二进制文件的 Python 包。
- 这是一个python package，这个python package的作用就是为了生成GGUF格式的二进制文件
- 上述给到的地址就是GGUF文件格式的规范

See convert_hf_to_gguf.py as an example for its usage.

- 请参阅 convert_hf_to_gguf.py 作为其用法的示例。
- 上述指出的python脚本就是llama.cpp项目顶层中的convert_hf_to_gguf.py文件

Installation

```
# 给特定版本的python安装gguf包（用来写入GGUF格式的二进制文件）
pip install gguf
```

API Examples/Simple Tools

examples/writer.py — Generates `example.gguf` in the current directory to demonstrate generating a GGUF file. Note that this file cannot be used as a model.（examples/writer.py在当前目录中生成一个example.gguf用来演示生成GGUF文件。注意：这个文件不能被用作一个model）

scripts/gguf_dump.py — Dumps a GGUF file's metadata to the console.（将一个GGUF文件格式的元数据输出到终端）

scripts/gguf_set_metadata.py — Allows changing simple metadata values in a GGUF file by key.（这个python脚本允许通过key来改变GGUF文件中的简单的元数据值）

scripts/gguf_convert_endian.py — Allows converting the endianness of GGUF files.（这个python脚本允许改变一个GGUF文件的大小端）

scripts/gguf_new_metadata.py — Copies a GGUF file with added/modified/removed metadata values.（这个python脚本允许复制包含添加/修改/删除的元数据值的GGUF文件）

Development

Maintainers who participate in development of this package are advised to install it in editable mode（建议参与此软件包开发的维护人员以可编辑模式安装它）:

- 想要参与这个gguf包的开发者的建议是：以可编辑的模式安装这个包

```
cd /path/to/llama.cpp/gguf-py
# 以editable mode安装gguf包
pip install --editable .
```

Note: This may require to upgrade your Pip installation, with a message saying that editable installation currently requires `setup.py`. In this case, upgrade Pip to the latest:

注意：这可能需要升级你的 Pip 安装，并显示一条消息，提示可编辑安装当前需要“setup.py”。在这种情况下，将 Pip 升级到最新版本：

```
# 使用pip安装升级后的pip
pip install --upgrade pip
```

Automatic publishing with CI（使用CI实现自动发布）

There's a GitHub workflow to make a release automatically upon creation of tags in a specified format.（GitHub 有一个工作流程，可以在创建指定格式的标签后自动发布。）

1. Bump the version in `pyproject.toml`.（修改 pyproject.toml 中的版本）
2. Create a tag named `gguf-vx.x.x` where `x.x.x` is the semantic version number.(创建一个名为“gguf-vx.x.x”的标签，其中“x.x.x”是语义版本号。)

   ```
   # 使用git创建一个名为gguf-v1.0.0的tag
   git tag -a gguf-v1.0.0 -m "Version 1.0 release"
   ```

3. Push the tags.（推送tag）

   ```
   git push origin --tags
   ```

Manual publishing（手动发布）

If you want to publish the package manually for any reason, you need to have `twine` and `build` installed（如果你以任何原因想要手动发布gguf包，那么你需要有twine、build工具）:

```
# 通过pip安装build、twine包
pip install build twine
```

Then, follow these steps to release a new version:

1. Bump the version in `pyproject.toml`.
2. Build the package:

   ```
   python -m build
   ```

3. Upload the generated distribution archives:

```
python -m twine upload dist/*
```

Run Unit Tests（运行单元测试）

From root of this repository you can run this command to run all the unit tests（从这个repo的根目录你可以运行这个命令从而运行所有的单元测试）

```
python -m unittest discover ./gguf-py -v
```

TODO

- ☐ Include conversion scripts as command line entry points in this package.（将转换脚本作为命令行入口点包含在此包中。）