Pull requests (for contributors)(PR:针对贡献者)

- Test your changes（测试你的变更）:
  - Using the commands in the `tests` folder. For instance, running the `./tests/test-backend-ops` command tests different backend implementations of the GGML library（使用 `tests` 文件夹中的命令。例如，运行 `./tests/test-backend-ops` 命令可测试 GGML 库的不同后端实现）
  - Execute the full CI locally on your machine before publishing（在发布之前执行相关命令）
- Please rate the complexity of your PR (i.e. `Review Complexity : Low`, `Review Complexity : Medium`, `Review Complexity : High`). This makes it easier for maintainers to triage the PRs.（请给PR进行投票，这使得维护人员更容易对 PR 进行分类）
  - The PR template has a series of review complexity checkboxes `[ ]` that you can mark as `[X]` for your convenience
- Consider allowing write access to your branch for faster review（考虑允许对你的分支进行写访问以便更快地进行审查）
- If your PR becomes stale, don't hesitate to ping the maintainers in the comments（如果你的 PR 变得陈旧，请随时在评论中联系维护者）

Pull requests (for collaborators)（PR：针对协作者）

- Squash-merge PRs（压缩合并 PR）
- Use the following format for the squashed commit title（压缩提交标题使用以下格式）: `<module> : <commit title> (#<issue_number>)`. For example: `utils : fix typo in utils.py (#1234)`
- Optionally, pick a `<module>` from here: https://github.com/ggerganov/llama.cpp/wiki/Modules

Coding guidelines（编码准则）

- Avoid adding third-party dependencies, extra files, extra headers, etc.（避免添加第三方库依赖，额外的文件、额外的headers等等）
- Always consider cross-compatibility with other operating systems and architectures（总是考虑不同操作系统、不同硬件架构之间的可移植性）
- Avoid fancy looking modern STL constructs, use basic `for` loops, avoid templates, keep it simple（避免使用看起来很fancy的现代STL构造器使用基础的for循环，避免模板，保持简单）
- There are no strict rules for the code style, but try to follow the patterns in the code (indentation, spaces, etc.). Vertical alignment makes things more readable and easier to batch edit（这里对于代码风格没有限制条款，但是请尝试在代码中遵循下列模式（对齐、空格等等）垂直对其将会使得代码更加的可读、更加容易批量编辑。）
- Clean-up any trailing whitespaces, use 4 spaces for indentation, brackets on the same line, `void * ptr`, `int & a`（清除所有尾随空格，使用 4 个空格缩进，括号放在同一行）
- Naming usually optimizes for common prefix (see https://github.com/ggerganov/ggml/pull/302#discussion_r1243240963)（命名通常针对通用前缀进行优化）
- Tensors store data in row-major order. We refer to dimension 0 as columns, 1 as rows, 2 as matrices（张量按行主序存储数据。我们将维度 0 称为列，维度 1 称为行，维度 2 称为矩阵）

- Matrix multiplication is unconventional（矩阵乘法是不常规的：和通常的理解是不同的，不过现在还可以不进行深入理解）: `C = ggml_mul_mat(ctx, A, B)` means $C^T = A B^T \Leftrightarrow C = B A^T.$