

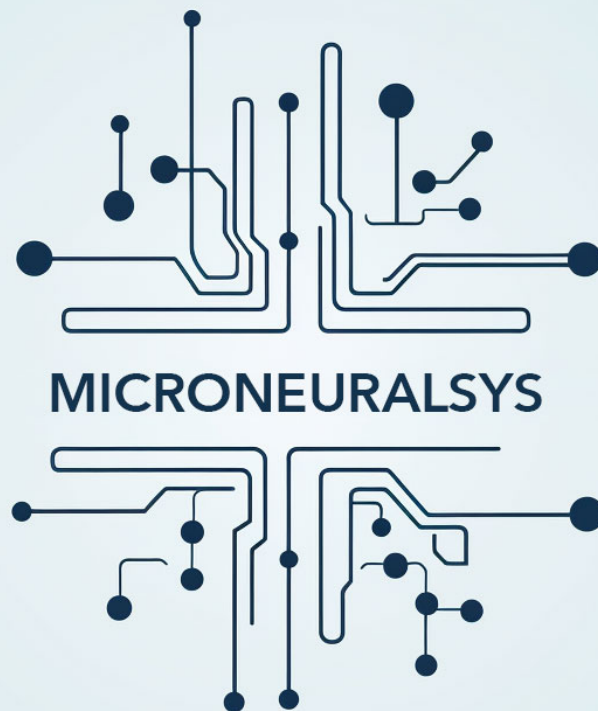
# Code Base和Git使用小记—— MicroNeuralSys

---

## 1. 仓库名/组名-.-, 初版Logo

---

MicroNeuralSys



## 2. 关于权限的说明

---

- 默认是member，组织内的成员对所有仓库具有可读、可使用权限，但没有相应的修改权限，只有组织的管理者和相应仓库的创建者拥有修改权限，或者可以让管理者、创建者给想要修改的成员添加相应的权限。

Navigation bar: <> Code Issues Pull requests Actions Projects Security Insights **Settings**

Left sidebar (under General):

- Access
  - Collaborators and teams**
- Code and automation
  - Branches
  - Tags
  - Rules
  - Actions
- Webhooks
- Pages
- Custom properties

Security:

- Code security and analysis
- Deploy keys
- Secrets and variables

Integrations:

- GitHub Apps

Main content area:

### Who has access

PRIVATE REPOSITORY: Only those with access to this repository can view it. [Manage](#)

BASE ROLE: All 5 members can access this repository. [Manage](#) (Read)

DIRECT ACCESS: 0 teams or members have access to this repository. Only [Owners](#) can contribute to this repository.

### Manage access

Create team


You haven't added any teams or people yet

Organization owners can manage individual and team access to the organization's repositories. Team maintainers can also manage a team's repository access. [Learn more about organization access](#)

[Add people](#) [Add teams](#)

Close button (X)

## Add people to AlexNet\_MLP\_PCA\_New

 **JL**  
n1012408214 Close (X)

### Choose a role

- ☐ **Read** (Base role)  
Recommended for non-code contributors who want to view or discuss your project.
- ☐ **Triage**  
Recommended for contributors who need to manage issues and pull requests without write access.
- ☐ **Write**  
Recommended for contributors who actively push to your project.
- ☐ **Maintain**  
Recommended for project managers who need to manage the repository without access to sensitive or destructive actions.
- ☒ **Admin**  
Recommended for people who need full access to the project, including sensitive and destructive actions like managing security or deleting a repository.

[Add n1012408214 to this repository](#)

- 组织内的所有成员都有权限创建自己的团队，团队成员可以自己邀请，可以是组织内的，也可以是组织外的，团队创建者可以给不同的团队成员设置不同的权限，可以理解为组织里面的组织。
- 团队创建的时候可以设置为私有或者公有，私有的话，即使一个组织内的人，也不可以看到。

# Seamless communication with teams

Teams are a great way for groups of people to communicate and work on code together. Take a look at why they're great.



## Flexible repository access

You can add repositories to your teams with more flexible levels of access (Admin, Write, Read).



## Request to join teams

Members can quickly request to join any team. An owner or team maintainer can approve the request.



## Team mentions

Use team @mentions (ex. @github/design for the entire team) in any comment, issue, or pull request.

[Learn more](#)

Find a team...

New team

Select all

Visibility

Members

2023 Master



3 members

0 teams

## Create new team

### Team name

You'll use this name to mention this team in conversations.

### Description

What is this team all about?

### Parent team

Select parent team

### Team visibility

☒ Visible Recommended

A visible team can be seen and [@mentioned](#) by every member of this organization.

☐ Secret

A secret team can only be seen by its members and may not be nested.

### Team notifications

☒ Enabled

Everyone will be notified when the team is @mentioned.

☐ Disabled

No one will receive notifications.

Create team

- 上述所有权限，可以创建完后进行修改，并不是一成不变。也就是说，member可以被组织创建者改为owner等等。
- 未完待续！

## 3. 新建仓库

- 没什么特殊需求，新建团队可以不用管。**平常上传资料或者代码的时候新建仓库即可。**
- 仓库我习惯使用git命令行提交，我下面给大家说下常用的几个命令，如果不出意外按照下面的输入就能正常提交，出现小问题可以自行查找资料。=\_=
- 在使用git前需要先下载，下面是Windows和Linux两种下载方式——两分钟下载完。=。=

## 1. 安装 Git

Git 是分布式版本控制系统，你需要先安装 Git。

### 在 Windows 上安装 Git

#### 1. 下载 Git:

- 访问 [Git 官方网站](#)，点击“Download for Windows”下载 Git 安装程序。

#### 2. 安装 Git:

- 运行下载的安装程序，按照默认设置进行安装。安装过程中可以选择是否安装 Git Bash（推荐安装），这将提供一个类似 Unix 的终端界面。

### 在 macOS 上安装 Git

#### 1. 使用 Homebrew 安装（推荐）：

- 如果你已经安装了 Homebrew，可以通过以下命令安装 Git：

```
1 | brew install git
```

#### 2. 通过 Git 官方网站下载并安装：

- 访问 [Git 官方网站](#)，下载适用于 macOS 的安装程序并运行安装。

### 在 Linux 上安装 Git

#### 1. 使用包管理器安装

- 在 Debian 或基于 Debian 的系统（如 Ubuntu）上：

```
1 | sudo apt update
2 | sudo apt install git
```

## 2. 配置 Git

安装完成后，你需要配置 Git 的用户信息。打开终端（Git Bash, macOS Terminal, Linux Terminal）并运行以下命令：

```
1 | git config --global user.name "Your Name"
2 | git config --global user.email "your.email@example.com"
```

### 3. 提示

其实上面名可以在IDE开发环境写，推荐VScode轻量化IDE。

### 4. 推自己的资料

#### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).

Owner \*

MicroNeuralSys

Repository name \*

AlexNet\_SBPCA

✓ AlexNet\_SBPCA is available.

Great repository names are short and memorable. Need inspiration? How about [automatic-rotary-phone](#) ?

Description (optional)

Repository use only within chuan group

☐



Public

Anyone on the internet can see this repository. You choose who can commit.

☒



Private

You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

You are creating a private repository in the MicroNeuralSys organization.

Create repository

- 记得选择Private，这样非组织成员就看不见了。

```
1 | git init // 初始化一个本地仓库
```

```
1 | git add . // 将全部文件加入到初始化的本地仓库里面
```

- 先进入对应文件夹，在git add .
- github在不配置git-lfs的前提下不让上传单个超过100M的大文件，总量超过100M应该是可以的
- 可以自行查找配置git-lfs
- ```
1 | git add 文件名 // 可以单个上传特定文件
```

```
1 | git commit "自己的描述"
```

```
1 | git checkout -b main //创建一个名为main的分支，并切换到该分支。
```

```
1 | git remote add 远程仓库别称 远程仓库地址
2 | 例子:
3 | git remote add origin https://github.com/MicroNeuralSys/AlexNet_SBPCA.git
4 | //自己可以起名字替换origin
```

```
1 | git push -u 起的远程仓库别称 所在的分支名
2 | // git push -u origin main
```

```
> git init
Reinitialized existing Git repository in /home/leo/ztl/AlexNet_SBPCA_myself/AlexNet_SBPCA/al
> git add .
> git commit -m "first AlexNet_SBPCA"
[main 140351a] first AlexNet_SBPCA
15 files changed, 513 insertions(+), 94 deletions(-)
rewrite Model/_pycache_/netBase.cpython-38.pyc (81%)
create mode 100644 New/_pycache_/sbpca_paper.cpython-38.pyc
create mode 100644 New/sbpca_paper.py
create mode 100644 New/test.py
create mode 100644 NewGlobalVar.py
rewrite README.md (97%)
rewrite __pycache__/GlobalVar.cpython-38.pyc (66%)
create mode 100644 __pycache__/NewGlobalVar.cpython-38.pyc
delete mode 100644 log/log_mnist_0.01_128_True_10_2024-05-08_15:13:14.txt
> git remote -v
origin https://github.com/future-Frank/AlexNet_MLP_SBPCA.git (fetch)
origin https://github.com/future-Frank/AlexNet_MLP_SBPCA.git (push)
purple_wisteria_ztl http://120.26.211.150:3000/WH_TL/AlexNet_mlp_SBPCA.git (fetch)
purple_wisteria_ztl http://120.26.211.150:3000/WH_TL/AlexNet_mlp_SBPCA.git (push)
> git remote remove origin
> git remote add Lab https://github.com/MicroNeuralSys/AlexNet_SBPCA.git
> git push -u Lab main
Enumerating objects: 198, done.
Counting objects: 100% (198/198), done.
Delta compression using up to 32 threads
Compressing objects: 100% (190/190), done.
Writing objects: 100% (198/198), 508.34 KiB | 6.35 MiB/s, done.
Total 198 (delta 75), reused 0 (delta 0)
remote: Resolving deltas: 100% (75/75), done.
To https://github.com/MicroNeuralSys/AlexNet_SBPCA.git
* [new branch]      main -> main
```

## 4. 上传文档 (windows终端, 上面我是用的Linux终端)

- 进入文件所在目录
- 鼠标右键->在终端打开

```
Windows PowerShell
版权所有 (C) Microsoft Corporation, 保留所有权利。

安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS E:\文档\文档\组会汇报> ls

目录: E:\文档\文档\组会汇报

Mode                LastWriteTime         Length Name
----                -
-a----             2024/4/1       15:47       3196457 2024.3.31组会汇报.pptx
-a----             2024/4/15       15:06       4938464 2024.4.15组会汇报.pptx
-a----             2024/4/22       10:40       2616807 2024.4.22组会汇报.pptx
-a----             2024/4/8        11:29       4850917 2024.4.7组会汇报.pptx
-a----             2024/5/13       10:28       5147345 2024.5.13组会汇报.pptx
-a----             2024/5/14       14:16         463326 Code Base和Git使用小记—MicroNeuralSys.pdf
-a----             2024/5/13       10:07       2093665 main.pdf
-a----             2024/5/13        9:55       1061873 闫恩硕师兄组会汇报.pptx

PS E:\文档\文档\组会汇报> |
```

```
PS E:\文档\文档\组会汇报> git init
Initialized empty Git repository in E:/文档/文档/组会汇报/.git/
PS E:\文档\文档\组会汇报> git add '.\Code Base和Git使用小记—MicroNeuralSys.pdf'
PS E:\文档\文档\组会汇报> git commit -m "使用小提示"
[master (root-commit) e7822c2] 使用小提示
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 "Code Base\345\222\214Git\344\275\277\347\224\250\345\260\217\350\256\260\342\200\224\342\200\224MicroNe
```

```
PS E:\文档\文档\组会汇报> git checkout -b weihao_version
Switched to a new branch 'weihao_version'
PS E:\文档\文档\组会汇报> git remote add lab https://github.com/MicroNeuralSys/Code-Base-Tips.git
```

```
PS E:\文档\文档\组会汇报> git checkout -b weihao_version
Switched to a new branch 'weihao_version'
PS E:\文档\文档\组会汇报> git remote add lab https://github.com/MicroNeuralSys/Code-Base-Tips.git
PS E:\文档\文档\组会汇报> git push -u lab weihao_version
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 32 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 426.82 KiB | 35.57 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MicroNeuralSys/Code-Base-Tips.git
 * [new branch]      weihao_version -> weihao_version
branch 'weihao_version' set up to track 'lab/weihao_version'.
```

- 当我第二次提交一个文档

```
PS E:\文档\文档\组会汇报> git push -u lab weihao_version
To https://github.com/MicroNeuralSys/Code-Base-Tips.git
! [rejected]        weihao_version -> weihao_version (fetch first)
error: failed to push some refs to 'https://github.com/MicroNeuralSys/Code-Base-Tips.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

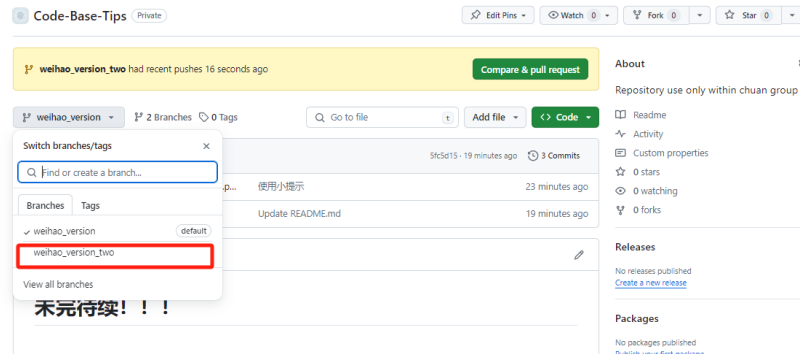
- 出现上述报错的原因是我提交的分支之间已经有了文件。

- 两个解决办法
- 第一种先pull 再 push
- 建议第二种

- ```
1 | git branch -M weihao_verison_two
2 | //强制更换当前分支名
```

```
PS E:\文档\文档\组会汇报> git branch -M weihao_version_two
PS E:\文档\文档\组会汇报> git push -u lab weihao_version_two
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 32 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 684.56 KiB | 29.76 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
remote:
remote: Create a pull request for 'weihao_version_two' on GitHub by visiting:
remote:   https://github.com/MicroNeuralSys/Code-Base-Tips/pull/new/weihao_version_two
remote:
To https://github.com/MicroNeuralSys/Code-Base-Tips.git
 * [new branch]      weihao_version_two -> weihao_version_two
branch 'weihao_version_two' set up to track 'lab/weihao_version_two'.
```

○



○ 这样可以避免文件冲突了

## PS. 建议

- 建议一个类型代码新建一个仓库。
- 至于文档，若公用可以放到我创建的Public Document仓库里，拉个新分支，即不要使用main分支名
- 个人资料，或者论文，建议新建自己的仓库。=.=

额，后面有什么需要补充的，我再更新，文档也已经上传到Github置顶，都可以补充上传覆盖之前的。