From 知乎用户：深度学习科研，如何高效进行代码和实验管理?

我本科的专业是软件工程，自从开始接触机器学习以来，越来越感觉到平时的实验代码其实很大程度上根本没有遵守软件工程的一些准则，或者说，深度学习破坏了软件工程，这一点在公司内部尤为明显。

经常需要面对的问题是，针对同一个任务，我们需要同时使用多个模型做多种实验，并在同一个模型上进行多种尝试，在这一过程中，如果不注意管理，就会导致代码越来越乱，文件夹越来越多，除了你自己，没人知道你的代码库之间有何种关系；更要命的是，如果你中间休息了几天没看，估计等你回来继续干的时候，自己也会弄混哪个是哪个了。

下面总结一下我自己的一些实践经验：

1. 使用一致的 job/project 命名习惯。
2. 在决定进行下一阶段的研发之前，重构一下当前的代码，为下一阶段的工作扫清隐患（如果你的部分代码需要复用）。
3. 实验的各个部分需要的超参数，最好进行统一的管理，以防每做一点参数更新，就需要更改多个文件。
4. 实验的输出，不管是 log 还是定量的分析图表，跟 job 命名一样，使用同一种习惯去整理。
5. 灵活地使用 git（我推荐使用 gitlab）。对于同一个任务，可以使用分支管理不同的模型；也方便在出现 bug 时跟以前的文件内容做对比；以及可以放置某个版本除了重大问题可以回退；除此之外，git 本身的 history 也是自己整理实验思路很好的参考。
6. 对于可视化，评估，backbone 等经常需要被复用的代码，可以整理成一个 module，尽量保证 task specific 的部分和 general 的部分分离开。
7. 保证代码的可读性，撰写必要的注释；一个很好的习惯是，在文件的开头或者其他什么地方，先写好当前文件的整体思路
8. 训练完成后对模型进行评估是一个比较繁琐的工作，可以设计一个自动化的流程，自己只需要更改一些参数配置，然后让程序自动完成当前训练的评估。
9. 对于实验中比较重要的一些结论，或者 idea，可以使用 markdown 整理成文档
10. 如果你需要同时在本地和集群进行开发和训练，那么设计一个合理的结构，从而可以以最少的改动可以在本地和集群上跑起来，也是很重要的。
11. overleaf 确实是一个很好用的东西，我觉得跟 git 类似的一点是，将自己的工作保存到云端，而不是本地，是一个风险更低，更灵活，高效的方式。