AI使用心得报告

经过一个学期的高强度代码研究，AI的优势和劣势也逐渐明晰。AI是一个非常好的集成了所有知识库的百科全书，他收集了绝大多数常见的问题，基础性的数据结构，优秀的代码规范和现代的语法。在另一方面他也只会这些，类似书里查不到的他就不知道，很多个性化的代码他就难以理解。尤其是在做代码重构的时候，如果和数据库的已有代码有所重合，他就很容易犯迷糊。

我们认为AI是明显没有认知的智能，他完全无法理解一段代码的意义。他最多根据名字和局部的代码读一个一知半解，缺乏真正的认知。文件一多，代码一长他就看不出来了。一个典型案例是我在编写测试案例的时候，AI给我编写的都没有遵循轮流下棋规则。这大概是他没有看到SwitchPlayer的函数吧。此外很多报的错误可能也比较少见，他都无法给出简要的解决方案或者清晰的指出要害所在。也有可能是他不知道该去查看些什么文件。

AI很擅长给我介绍我没学过的东西，同时帮我解释系统的报错，给我代码不太符合规范或者存在小风险的地方提出修改建议。他总是给我写出命名规范，格式清爽的函数，还有风一般的打字速度，能很大的便利我写代码的过程。但受到神经网络的限制，就像一个学而不思的人，目前难以走向深刻的创造。编程过程中还是需要有强大的心力做好代码的主干思路，并将细枝末节的重复劳动交给AI处理。

在这里想特别提一下Copilot的代码补全功能。不得不承认这是一个无比强大的功能，大大提高了生产力。他的生成比普通的编程LLM快了大概五到十倍，所付出的代价就是他只会读很少的一段上下文。再加之其不需要额外输入的prompt，他就像我们的输入法提示语一样，有效但不是很可靠。往往会带来额外的debug时间。作为一个好的代码习惯，还是要仔细阅读其中生成的每一行代码（copilot如果连续生成了几行后，有概率一次生成一大段），最好是每一行确认之前都要想清楚是否是心里想要的。

有了AI之后很多数据结构和算法不再成为一些硬的知识壁垒，对于真正有心学习的人来说，可以很快的上手，也可以在AI的帮助下写出一些简易的代码。只要他们有对代码的理解和学习能力，有很多成品的代码在实践中学习是非常高效的。我们也不再需要那些基础的代码速度，背一堆库的名字和语法。大概差不多知道长什么样就可以，但是对理解其中的逻辑和原理就显得格外重要。不然AI写出来的高级代码甚至看不懂，也不会用不会改。本质上来说，AI节约了人类做简单重复工作的时间，那整体生产力提升了，AI时代下的程序员就需要在AI的帮助下提升自己的生产力。在对代码记忆本身要求更低的情况下，要提升的就是数据结构的理解，代码框架的构建，各种算法和逻辑之间的选择。另一方面，正如课上说的“每个程序员都能写出编译器看得懂的代码，但是只有好的程序员才可以写出人类看得懂的代码”，AI可以面向人类学习，但是人类没有这么多精力。写好Human- friendly的代码（包括但不限于阅读友好，修改友好，测试友好）也成为了更重要的环节，千万不能让AI提升的效率，全浪费在了阅读，修改别人莫名其妙的代码了。我也曾打开检查器看过腾讯QQ的一个抽卡功能网页的内置html，里面写的东西在不同版本之间命名规律完全不一致，很多变量也不知道有什么用，我觉得避免这样的内容在AI时代就更重要，AI也可以帮我们更好的重构之前写过的不够完善的代码。

总的来说，AI更接近一个听得懂人话的知识库，人类要学会的则是更硬核的功夫。