La Prochaine Direction des Éditeurs de Code IA

Récemment, je travaillais sur l'ajout d'un pipeline `xelatex` à GitHub Actions.

J'ai rencontré un problème avec le package `fontawesome5` dans le flux GitHub. La solution proposée par 4o-mini (installer TeX Live 2021 et utiliser `tlmgr install fontawesome5`) n'a pas fonctionné pour moi. Cependant, 4o a suggéré une meilleure approche : passer à TeX Live 2023 et toujours utiliser `tlmgr` pour installer `fontawesome5`. Bien que cela n'ait pas complètement résolu le problème, le passage à TeX Live 2023 a considérablement amélioré la situation.

J'ai utilisé ChatGPT pour m'aider à résoudre le problème. Pour plus de détails, consultez Ce que ChatGPT O1 peut faire que 4o-mini ne peut pas.

À ce stade, je n'utilisais pas d'éditeurs comme Cursor ou Windsurf, bien que je les aie essayés dans un autre projet. Le problème avec ces éditeurs de code est qu'ils ne captent que les sorties de tests locaux, ce qui limite leur fonctionnalité dans les environnements cloud.

Dans les workflows tels que GitHub Actions, les jobs Jenkins, ou tout flux de déploiement ou de test de code, les éditeurs de code doivent être mieux intégrés. Ils devraient offrir une interaction fluide avec le cloud et les processus CI/CD.

Cette intégration s'applique également à d'autres outils de création de contenu, qu'il s'agisse de texte, d'images, d'audio ou de vidéo. Ces outils devraient être intégrés avec des systèmes de tests A/B. Les outils d'IA pourraient générer du contenu, et les outils de tests A/B pourraient fournir des retours. Cette dynamique est similaire à l'apprentissage par renforcement basé sur les retours humains (RLHF), où les modèles d'IA s'améliorent au fil du temps en fonction des retours du monde réel.

Cette idée d'étendre le RLHF (Apprentissage par Renforcement à partir de la Rétroaction Humaine) au-delà des simples sorties de modèles, en l'intégrant dans des environnements de test et de déploiement réels, semble être une direction prometteuse pour améliorer à la fois les éditeurs de code et les outils de création de contenu pilotés par l'IA.

Le test peut être instantané ou long, et il peut être soit automatisé soit assisté par des humains. Si les tests sont automatisés, comme les tests A/B utilisateur pour un outil d'IA, ils impliquent toujours un retour humain, mais le processus est automatisé. Par exemple, nous pouvons laisser l'ordinateur vérifier les résultats chaque jour ou chaque heure en fonction des résultats des tests A/B pour améliorer le processus de création. De même, pour les tâches Jenkins ou GitHub Actions, nous pouvons laisser l'ordinateur vérifier après que leurs tâches sont terminées.

Si une assistance humaine est impliquée, la machine ne peut pas pleinement comprendre le feedback, qui est souvent un peu vague. Par exemple, lorsque des outils d'IA créent du contenu comme des images ou des vidéos, les humains pourraient souligner que le contenu n'est pas assez drôle ou qu'un détail spécifique devrait être amélioré. Les machines ont encore un long chemin à parcourir pour rendre tout parfait, et ce qui est considéré comme "parfait" est souvent subjectif, dépendant des goûts individuels. C'est le feedback humain qui aide à améliorer les choses.

En théorie, toutes les règles définies par l'homme peuvent être écrites sous forme de prompts. Il existe des prompts utilisateur et des prompts système. Nous devrions nous concentrer sur l'amélioration des prompts plutôt que de corriger la sortie à chaque fois.