UNIVERSITÉ Grenoble Alpes

Projet Master 1 IC

Titre du sujet : U-QUAIL — plateforme d'évaluation par des experts des réponses produites par des IA sur graphes de connaissances biomédicales

Nom de l'encadrant : Sandrine Muller — Chaire de Professeur Junior « IA pour la santé », Laboratoire d'Informatique de Grenoble (LIG) / UFR de pharmacie, Université Grenoble Alpes
Email de l'encadrant : …… sandrine.muller@univ-grenoble-alpes.fr
Téléphone de l'encadrant : …… +33 4 57 42 14 18

1. Présentation du sujet

(merci de nous faire parvenir le document dans un format éditable)

Contexte. Les moteurs de raisonnement sur graphes de connaissances biomédicales — comme le Biomedical Data Translator, programme financé par le NIH auquel participe notre équipe — répondent à des questions du type « quels gènes pourraient être impliqués dans cette maladie ? » en parcourant des millions de relations agrégées depuis des dizaines de bases publiques. Le projet U-QUAIL construit une échelle d'évaluation adaptée à cet usage : chaque réponse est jugée sur trois dimensions — qualité du raisonnement, usage des sources de données, qualité du résultat — par des experts du domaine.

Le besoin, et ce qui manque aujourd'hui. La partie « calcul » du benchmark (sélection des paires question/réponse à partir de la structure du graphe) est en cours dans l'équipe. Ce qui manque, et qui fait l'objet de ce projet, c'est l'outil permettant de collecter les jugements d'experts de façon rigoureuse et reproductible. Ces annotations se font actuellement dans des feuilles de calcul : pas de randomisation, pas de traçabilité, pas de mesure d'accord entre annotateurs — les résultats sont donc inexploitables pour une publication. Une tentative précédente reposait sur une plateforme tierce, abandonnée parce qu'incompatible avec les règles européennes de protection des données. Une instance PYBOSSA — logiciel libre de micro-tâches, dans le fork maintenu par Bloomberg — sera fournie comme socle sur une VM de Perseus : le projet consiste à l'étendre et à l'orchestrer, non à repartir de zéro.

Travail demandé. Étendre et orchestrer une plateforme d'annotation experte construite sur PYBOSSA (Python/Flask), de la spécification à la mise en service. Le socle fournit les comptes annotateurs, la distribution des tâches une par une — un annotateur ne parcourt jamais le lot complet — et la redondance. Le projet porte sur ce qui manque au-dessus : le plan expérimental, l'interface d'annotation et la supervision des campagnes.

1. Générateur de plan d'assignation. À partir des lots de questions/réponses fournis par l'équipe (JSON/CSV, avec strates et métadonnées) et l'attribution des questions/réponses par par micro-lots, produire un plan annotateur × item × position : recouvrement contrôlé (chaque item vu par k annotateurs, randomisation de l'ordre, présentation aveugle du modèle ayant produit la réponse, budget par annotateur, réallocation en cas d'abandon. Plan versionné et rejouable (graine aléatoire consignée) : c'est la condition de reproductibilité pour la publication.
2. Intégration au socle PYBOSSA. Création des projets et des tâches via l'API à partir du plan, configuration de la redondance et de l'ordonnanceur, import et export automatisés.

UNIVERSITÉ Grenoble Alpes

3. Interface d'annotation (task presenter JavaScript). Question, réponse et chemin de raisonnement présentés côte à côte basé sur la représentation TRAPI 2.0, échelle U-QUAIL à trois dimensions, reprise de session. Lecture du graphe pour afficher la provenance des chemins de raisonnement ayant conduit à chaque réponse.
4. Base de supervision et tableau de bord multi-campagnes. Une base PostgreSQL propre au projet (campagnes, items, assignations, annotations), alimentée par synchronisation depuis l'API PYBOSSA ; tableau de bord : avancement par campagne et par annotateur, accord inter-annotateurs (kappa de Cohen, alpha de Krippendorff, et méthodologie développée dans l'équipe), temps passé par item, taux de réussite aux items de contrôle, export des données annotées dans un format versionné et documenté.
5. Couche d'abstraction du socle. Les accès à PYBOSSA sont isolés derrière une interface d'adaptation, afin que la plateforme d'annotation reste remplaçable sans réécrire l'orchestration.
6. Déploiement et conformité. Déploiement reproductible (Docker et docker-compose), authentification, journalisation, conformité RGPD : hébergement européen, consentement des annotateurs, anonymisation, aucun service de mesure d'audience tiers, et indépendance des annotateurs — aucun accès aux jugements des autres avant de rendre les siens.

Intérêt pour les étudiants. Le projet couvre un cycle complet : recueil du besoin auprès de vrais utilisateurs (chercheurs en biologie et en informatique, en France et aux États-Unis), conception, développement, tests, déploiement, puis mise en service réelle avec une première campagne d'annotation avant la fin du projet. Il combine une partie algorithmique — le plan d'assignation est un plan en blocs incomplets, sous contraintes d'équilibrage et de randomisation —, une partie interface et une partie qualité des données ; c'est avant tout un sujet d'interaction homme-machine, de modélisation de données et de qualité logicielle. Partir d'un logiciel libre existant plutôt que d'une application neuve est en soi formateur : lecture de code tiers, extension par les points prévus à cet effet, contribution en amont possible. L'outil servira à une publication scientifique à laquelle les étudiants seront associés, et il a vocation à être diffusé en logiciel libre. Aucune connaissance en biologie n'est requise : le vocabulaire nécessaire sera fourni.

Livrables. Cahier des charges et maquettes ; générateur de plan d'assignation documenté et testé ; task presenter et extensions PYBOSSA ; base de supervision et tableau de bord ; déploiement reproductible (Docker) ; jeu de tests automatisés ; documentation utilisateur et technique ; dépôt Git ouvert ; rapport d'une première campagne d'annotation.

2. Références

— Biomedical Data Translator (NCATS/NIH), programme et outils : https://ncats.nih.gov/research/research-activities/translator
— Biolink Model, schéma commun des graphes de connaissances biomédicaux : https://biolink.github.io/biolink-model/
— ARAX / RTX-KG2, moteur de raisonnement et graphe utilisés dans le projet : https://github.com/RTXteam
— https://pmc.ncbi.nlm.nih.gov/articles/PMC11177514/
— https://arxiv.org/abs/2407.08940
— https://arxiv.org/abs/2505.04651
— https://pubs.rsc.org/en/content/articlelanding/2024/dd/d3dd00185g
— Mesures d'accord inter-annotateurs : kappa de Cohen, alpha de Krippendorff, + methodologie in house (documentation de référence fournie au démarrage).
— PYBOSSA, socle de micro-tâches retenu pour le projet, fork maintenu par Bloomberg : https://github.com/bloomberg/pybossa

UNIVERSITÉ Grenoble Alpes

— Documentation PYBOSSA (task presenter, API, ordonnanceur) : https://docs.pybossa.com/
— Artstein, R. & Poesio, M. (2008). Inter-Coder Agreement for Computational Linguistics. Computational Linguistics, 34(4), 555-596 : https://doi.org/10.1162/coli.07-034-R2

3. Positionnement du sujet

- Indiquez le niveau d'innovation du sujet proposé

Très innovant	❒	**X**	❒	❒	❒	Classique

- Indiquez la disponibilité de la documentation relative aux technologies à mettre en œuvre

Beaucoup	❒	**X**	❒	❒	❒	Aucune

- Indiquez le niveau d'abstraction du sujet

Théorique	❒	❒	❒	**X**	❒	Pratique

- Indiquez la quantité de développement à réaliser

Beaucoup	❒	**X**	❒	❒	❒	Peu

- Indiquez le niveau de difficulté des algorithmes à mettre en œuvre

Difficile	❒	❒	**X**	❒	❒	Facile

- Indiquez le niveau d'interaction avec d'autres composants logiciels

Ecosystème complexe	❒	**X**	❒	❒	❒	Application seule

- Indiquez le nombre d'étudiants souhaités pour le projet : 2 (front-end/back-end)

- Indiquez les langages et technologies à utiliser:
• Socle d'annotation : PYBOSSA (fork Bloomberg), Python/Flask — instance déployée et administrée par l'encadrement, fournie au démarrage du projet.
• Interface d'annotation : JavaScript/TypeScript (task presenter PYBOSSA) ; visualisation des chemins du graphe (Cytoscape.js ou D3).
• Orchestration et supervision : Python 3.11+, SQLAlchemy, PostgreSQL ; API PYBOSSA.
• Tableau de bord : Python (FastAPI et front léger, ou Streamlit) ; calculs d'accord inter-annotateurs en Python.
• Graphe de connaissances : Neo4j / Cypher, en lecture seule (une instance de démonstration sera fournie).
• Outillage : Docker et docker-compose, Git/GitLab, pytest, intégration continue légère.
• Contrainte de déploiement : hébergement dans l'Union européenne, sans service tiers de mesure d'audience.
• Les étudiants n'ont besoin de connaître ni PYBOSSA ni Neo4j au départ : la prise en main fait partie du projet et sera accompagnée.

4. Encadrement

- Combien de temps pouvez-vous consacrer à l'encadrement de projets chaque semaine ? Environ 1 h 30 par semaine : une réunion de suivi hebdomadaire en visioconférence, complétée par une relecture asynchrone des livrables (maquettes, code, documents). Les étudiants seront invités à présenter au groupe durant nos meetings hebdomadaires du Mardi matin (10h-11h). Un ingénieur d'étude travaillant sur le projet (orienté backend) sera disponible pour encadrer les étudiants à leur demande.

- Indiquez vos contraintes quant à l'encadrement

• Les étudiants de master devront être présents en personne au LIG lorsqu'ils travaillent sur le projet. Ils pourront aussi participer aux réunions de manière hybride en fonction de leur emploi du temps et intérêt.
• Je participe à des réunions de consortium en fin de journée et je me déplace une fois par an aux États-Unis : une semaine d'indisponibilité est à prévoir début Juin, annoncée à l'avance.
• Je suis enseignante-chercheuse, pas développeuse à temps plein : l'accompagnement technique au quotidien devra être assuré par l'enseignant référent ou des étudiants capables de travailler en autonomie. J'interviens sur les choix d'implémentation :

1. contraintes techniques pouvant influencer l'intégration, la reproducibilité, et le partage (science ouverte),
2. besoin,
3. les priorités,
4. la validation fonctionnelle et la mise en relation avec les utilisateurs.

- Indiquez vos contraintes quant au sujet proposé

• Données : uniquement des données publiques issues de graphes de connaissances biomédicaux ouverts. Aucune donnée de patient n'est manipulée.
• Les annotateurs sont des chercheurs volontaires : consentement explicite, anonymisation des jugements, hébergement des données dans l'Union européenne, aucun service de mesure d'audience tiers. Ce point est structurant, une première version du projet ayant dû être abandonnée pour cette raison.
• Code publié sous licence libre (MIT, Apache 2.0 ou openGL v3) et reproductible (dépôt Git, Docker). Commentaires en anglais uniquement. End user documentation explicite.
• Une partie des échanges avec les utilisateurs se fera en anglais ; les réunions d'encadrement et les livrables peuvent rester en français.
• Le format d'export doit rester compatible avec le pipeline existant (Python, Neo4j/Cypher, Biolink Model). Les spécifications d'interface seront fournies au démarrage.
• Le pipeline de génération du benchmark (calculs sur le graphe) n'est pas dans le périmètre du projet : il est développé en parallèle par l'équipe et fournira les jeux de données.
• L'instance PYBOSSA et la machine virtuelle qui l'héberge sont fournies et administrées par l'encadrement : les étudiants livrent un déploiement reproductible, mais ne portent pas la responsabilité d'exploitation (sécurité, sauvegardes, certificats, données personnelles).
• Contrainte scientifique structurante : l'indépendance des annotateurs. Un annotateur ne doit à aucun moment accéder aux jugements des autres avant de rendre les siens — critère qui a conduit à retenir PYBOSSA plutôt qu'un outil dont l'édition libre ne cloisonne pas les utilisateurs.

A retourner à :

Damien.Pellier@univ-grenoble-alpes.fr