Construction de graphe de connaissances avec des LLM

Mots-clés : LLM, bases de données graphe (Neo4j)

Lieu : LIP6 (Sorbonne Université), Équipe Bases de Données http://www-bd.lip6.fr/)
Encadrants du stage :

- Camelia CONSTANTIN (camelia.constantin@lip6.fr, LIP6-Équipe BD)
- Raphaël FOURNIER-S'NIEHOTTA (Raphael.Fournier@lip6.fr, LIP6-Équipe ComplexNetworks)

Contexte: Un graphe de connaissance (GC) est une structure interconnectée qui organise l'information en représentant des entités, leurs attributs et leurs relations. Les nœuds correspondent aux entités (par exemple « Napoléon Ier »), les arêtes définissent les interactions (comme « A_COMMANDÉ »), et les étiquettes et propriétés enrichissent ces nœuds (ex. *Empereur*, AnnéeDeCouronnement: 1804). Cette approche relationnelle permet d'intégrer des données issues de sources variées et de révéler des interconnexions complexes. Par exemple, dans un corpus historique, un GC peut relier deux biographies via un nœud commun « Bataille d'Austerlitz », facilitant l'analyse des liens entre figures historiques et événements.

Objectif : La construction d'un GC à partir de texte structuré ou non structuré permet de transformer des données dispersées en une représentation sémantique exploitable pour le raisonnement et la recherche avancée. Le processus classique inclut l'identification des entités, la résolution des ambiguïtés (références multiples, pronoms) et la création des relations qui structurent le graphe.

Apport des LLM: Les modèles de langage de grande taille (LLM) ont révolutionné cette tâche en remplaçant les pipelines rigides par des approches génératives. Grâce à leur entraînement sur des corpus massifs, les LLM peuvent synthétiser des représentations structurées directement à partir de texte brut, surmontant les limites des méthodes traditionnelles (manque de données, rigidité des ontologies). Ils permettent une extraction flexible, guidée par des invités (prompt engineering), avec des techniques comme Few-Shot, Chain-of-Thought ou Retrieval-Augmented Prompting pour améliorer la précision et la fiabilité. Les approches par agents LLM attribuent des rôles spécialisés à un ou plusieurs modèles pour exécuter des sous-tâches, combinant précision et efficacité.

Travail à réaliser: Le stage a pour objectif de construire un graphe de connaissance à partir des données historiques de **Studium** (http://studium.univ-paris1.fr) en utilisant des modèles de langage de grande taille (LLM). Cette base regroupe environ 15 000 fiches consacrées aux membres des écoles et de l'Université de Paris entre le XIIe et le XVIe siècle, contenant des informations biographiques et bibliographiques sur les professeurs, étudiants et autres acteurs universitaires.

Le projet débutera par la définition des types d'entités et de relations à extraire, ainsi que le choix entre une approche sans schéma ou guidée par une ontologie. L'approche sans schéma privilégie la génération directe des nœuds et des arêtes avant une normalisation en post-traitement, tandis que l'approche guidée par ontologie impose une structuration sémantique plus stricte grâce à des invités adaptées et des exemples few-shot.

L'extraction du graphe se fera en deux étapes. La première consiste à générer un ensemble unique d'entités à partir du texte en exploitant des LLM et des techniques d'ingénierie d'invités pour améliorer la couverture et la précision, notamment avec des stratégies comme Few-Shot et Auto-Réflexion pour détecter les entités manquées. La seconde étape établira les relations entre ces entités, en utilisant des approches comme Chain-of-Thought pour garantir la qualité des liens. Selon la nature des données, la prédiction des relations pourra s'appuyer sur des modèles de classification (pour des ensembles fixes) ou sur des méthodes de génération séquentielle (pour des corpus plus larges), avec des mécanismes pour corriger le déséquilibre entre relations présentes et absentes.

Une fois les entités et relations extraites, le graphe sera structuré et normalisé. Si l'approche retenue est guidée par schéma, une ontologie OWL pourra être générée pour produire un graphe RDF. Dans le cas d'une approche sans schéma, la normalisation reposera sur des techniques de liaison d'entités avec des bases comme DBpedia. Les instances seront agrégées pour former un graphe consolidé, en fusionnant les descriptions et en attribuant des poids aux arêtes selon leur fréquence.

Le contrôle qualité consistera à vérifier la cohérence, éliminer les doublons et tester les requêtes. Cette étape utilisera des langages comme SPARQL pour RDF ou Cypher pour Neo4j afin de garantir la fiabilité des données et d'évaluer la performance avec des métriques telles que la précision, le rappel et le F1-score.

Enfin, le graphe sera déployé et exploité dans une base orientée graphes comme Neo4j, avec des outils de visualisation tels que Neo4j Bloom pour faciliter l'exploration et l'analyse. Des résumés hiérarchiques pourront être générés pour optimiser l'intégration dans des systèmes de question-réponse ou d'aide à la décision.

Le graphe obtenu sera comparé à celui construit avec **Neo4j LLM Knowledge Graph Builder** (https://llm-graph-builder.neo4jlabs.com), un système qui utilise des LLM pour transformer des contenus variés tels que PDF, documents, images, pages web et transcriptions vidéo en deux graphes : un graphe lexical des documents et segments avec leurs embeddings, et un graphe d'entités reliant les nœuds et leurs relations, tous deux stockés dans Neo4j.

Compétences souhaitées: les élèves devront maîtriser plusieurs domaines techniques. Une solide compétence en **programmation Python** est indispensable pour le traitement des données, l'automatisation des tâches et l'intégration des API des modèles de langage. Une bonne connaissance des **modèles de langage (LLM)** et des techniques de **NLP** est requise, notamment l'ingénierie d'invités (prompt engineering), l'extraction d'entités et de relations, ainsi que l'utilisation de bibliothèques comme HuggingFace ou OpenAI API.

La compréhension des **technologies de graphes** est essentielle, incluant **Neo4j**, le langage de requêtes **Cypher,** et les standards RDF/SPARQL pour la structuration sémantique. Les élèves devront également être familiers avec les **ontologies** (OWL, RDF).