CentraleSupélec

Rapport de démarche

Multi Model Agent Translator

22 mai 2025

Table des matières

Introduction

Ce rapport de démarche vise à expliquer et justifier les choix techniques que nous avons faits au cours de ce projet. Le but est de permettre de mieux comprendre comment notre équipe en est arrivée à ce rendu final et de rendre compte des pistes qui se sont révélées infructueuses pour faciliter le travail de celui qui reprendra notre projet. Pour étoffer ce rapport, nous présentons aussi des pistes d'amélioration qui nous semblent pertinentes.

1 Contexte

1.1 Contexte du projet

Le projet s'inscrit dans le cadre d'une collaboration avec SOCOTEC Monitoring qui est une branche de SOCOTEC, une grande entreprise de génie civil. La branche Monitoring s'occupe de développer des capteurs, de récupérer leurs données et de les exploiter.

Monsieur Jacquet est notre interlocuteur chez SOCOTEC Monitoring, il travaille principalement sur les aspects data et logiciel des projets.

Le client souhaite développer un agent conversationnel qui s'appelle Bluegen et qui vise à améliorer la productivité des employés de SOCOTEC en automatisant certaines tâches.

1.2 Problème posé

L'objectif principal du projet est de développer une preuve de concept capable de traduire des documents professionnels en respectant la mise en forme et en améliorant la précision de la traduction des termes techniques du génie civil. Nous chercherons à surpasser les performances de DeepL en matière de traduction technique, notamment en utilisant la métrique COMET pour évaluer la qualité de notre modèle.

2 Outils et données mises à disposition

Une des contraintes du projet était de s'appuyer sur un LLM pour effectuer la traduction. Nous remercions Monsieur Jacquet et SOCOTEC qui nous ont fourni un accès aux serveurs Amazon pour effectuer des requêtes au modèle Llama 3 (70B). Nous devions travailler avec les outils open source LangGraph et LangChain. Monsieur Jacquet nous a aussi proposé plusieurs ressources afin de nous former sur le sujet.

Monsieur Jacquet nous a fourni deux fichiers PDF qui nécessitent typiquement d'être traduits par SOCOTEC. De plus, il nous a fourni la traduction automatique de ce fichier faite par Google Traduction.

À notre demande, notre client nous a fourni un glossaire au format PDF contenant les traductions anglais/français de termes techniques du génie civil, dont nous avons pu tirer profit dans le projet comme expliqué plus bas.

3 La pertinence des LLM dans le contexte de la traduction

L'utilisation des LLM était de fait une des contraintes du projet, et cela se justifie puisque, d'après la littérature scientifique, les LLM sont de bons outils de traduction.

Beaucoup d'outils de traduction commerciaux utilisent des réseaux de neurones composés de milliards de paramètres, comme DeepL, dont le modèle a été entraîné avec des transformeurs sur la base de données Linguee [1].

Les LLM tentent de prédire la suite la plus probable d'une certaine entrée. Ils ne sont donc pas conçus spécifiquement pour la traduction et peuvent effectuer des tâches très diverses, que ce soit pour des chatbots, de la synthèse de documents, de l'aide à la programmation, ou encore de la veille concurrentielle... Mais les LLM offrent des performances très intéressantes pour des applications de traduction. Une étude [2] s'est proposée de comparer la qualité de traduction de Google, DeepL, Tencent, GPT-3.5 et GPT-4. La méthodologie consiste à demander à des humains d'évaluer la qualité générale de différentes traductions selon un barème précis allant de 0 à 5, puis d'en faire une moyenne. Les résultats obtenus sont présentés en figure 1. On met ici en évidence que les LLM peuvent mieux performer que les outils traditionnels.

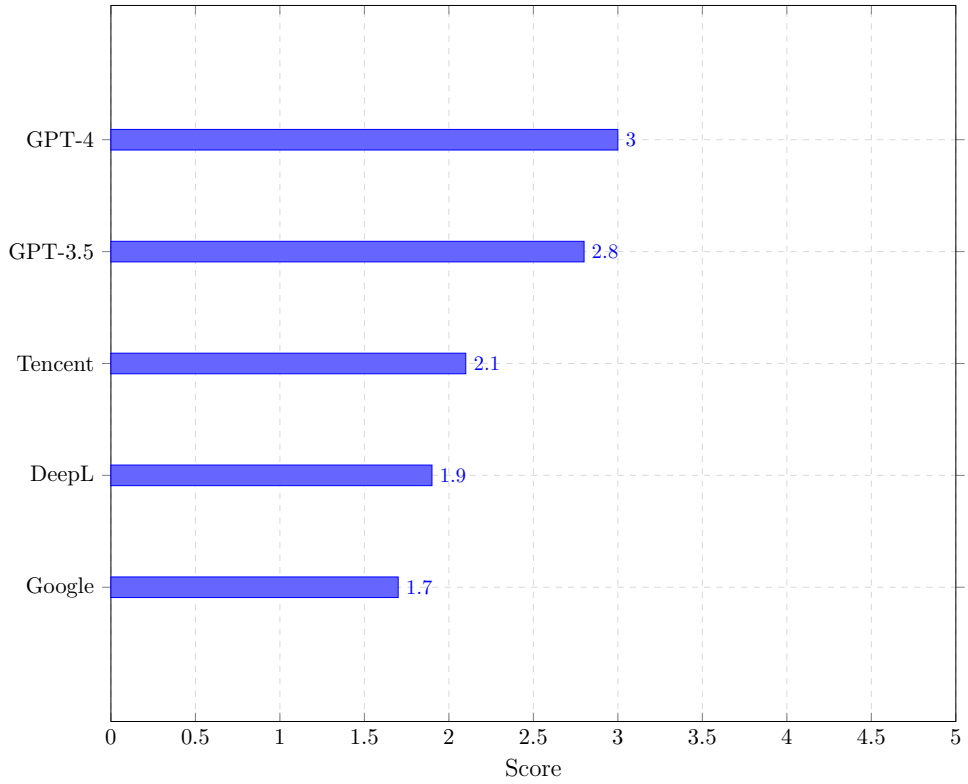

FIGURE 1 – Comparaison des scores de qualité générale des outils de traduction.

4 Extraction des données des fichiers

Le cahier des charges de notre projet imposait de pouvoir traiter des fichiers Word et PDF. Nous avons commencé par traiter les fichiers Word avec la librairie python-docx. Cependant, cette librairie présente plusieurs limitations techniques :

— La librairie permet effectivement de lire le contenu du fichier Word, mais elle découpe de façon arbitraire les phrases et les paragraphes. Il faut donc rassembler les bouts de texte pour que le LLM puisse tirer profit du contexte de la phrase pour améliorer la traduction, ce qui complexifie le code.

— De plus, python-docx ne permet pas de gérer les images contenues dans le fichier de façon simple ; il aurait fallu se reposer sur le fork d'un utilisateur et la solution n'était pas très robuste.

Par la suite, nous avons utilisé PyMuPDF pour traiter les documents PDF, et cet outil s'est révélé beaucoup plus performant pour conserver la mise en page et, en particulier, traiter les images.

La solution finalement retenue pour traiter les documents Word consiste à les convertir au format PDF, à appliquer la pipeline LangGraph adéquate, puis à les reconvertir au format Word, ce qui donne un résultat satisfaisant.

5 Prompting

5.1 Premiers prompts

Après quelques essais, nous sommes arrivés à un prompt qui se décompose en plusieurs parties. Une première partie indique la tâche à réaliser (traduction), puis on ajoute des instructions très précises pour obtenir un résultat qui n'ajoute pas de mots, qui laisse inchangés les liens, etc.

On ajoute une section qui contient le texte qui précède et qui succède, afin de donner du contexte au LLM, en espérant que cela améliore la qualité de la traduction.

Prompt de traduction

Predict the continuation of the translation into {target_language} for the current original chunk. Use the context so that the result concatenated with the previous translated chunk forms a coherent sentence.
IMPORTANT :
- Keep the approximate word count equal or lower.
- Detect if the chunk is part of a larger sentence and adjust accordingly.
- Use the last 20 words of the previous chunk and the first 20 words of the next chunk for context.
- If no logical continuation is possible, translate the chunk as a standalone sentence.
- Preserve non-translatable elements (links, emails, phone numbers, names, specialized terms) exactly.
- Preserve the original formatting (bullet points, numbered lists, headings, indentations).
CONTEXT :
Previous Translated Chunk : "{prev_translated_escaped}"
Previous Original Chunk (last 20 words) : "{previous_original_chunk}"
Current Chunk : "{current_chunk}"
Next Original Chunk (first 20 words) : "{next_original_chunk}"
OUTPUT ONLY the translated version of the current chunk.

5.2 RAG

Une des demandes très importantes de notre client était de traduire correctement les termes techniques contenus dans les fichiers, ce qui pose problème avec les outils actuels. Nous nous sommes appuyés sur un article scientifique [3] qui s'attaque à ce problème et propose trois techniques utilisables conjointement pour améliorer la qualité de traduction sur des domaines spécifiques en utilisant un LLM :

1. Faire du fine-tuning sur une base de données contenant des exemples de fichiers et leur traduction par un expert
2. Ajouter au prompt des exemples de phrases traduites qui sont proches de la phrase à traduire
3. Remplacer directement dans la phrase à traduire les termes techniques par leur traduction issue d'un glossaire et demander au LLM de conserver ces traductions préalables

Nous n'avons pas eu le temps de mettre en place la technique 1, mais nous aurions pu télécharger un petit modèle avec Ollama et le fine-tuner sur le glossaire. La solution 2 n'était pas envisageable du fait du manque de données ; par ailleurs, plus d'exemples de fichiers traduits auraient pu être mis à profit pour le fine-tuning.

Finalement, la solution 3 était la plus prometteuse, d'où notre demande à Monsieur Jacquet d'avoir un glossaire de génie civil. L'approche naïve consisterait à avoir une liste de couples *terme original : terme traduit*, cependant, le glossaire ne permet pas d'extraire une telle base de données aussi facilement du fait de sa mise en forme.

Extrait du glossaire

visite, passerelle f de inspection walkway
visite, trou m de manhole
visser vb screw [fix with a]
visser vb à fond screw home
visuel adj visual
visuel, examen m visual examination

Deux problèmes se posent :

1. Malgré un prétraitement de notre part, nous ne parvenons pas à prendre en compte toutes les particularités de la mise en forme pour obtenir une liste de couples *terme original : terme traduit*.
2. De plus, une simple recherche regex risque de laisser passer beaucoup de termes techniques du fait de potentielles typographies, accentuations, espacements...

Pour pallier ce problème, nous nous sommes appuyés sur les embeddings. Nous précalculons les embeddings des termes du glossaire. Puis, pour traduire une phrase, nous calculons d'abord son embedding, puis nous faisons une recherche des n plus proches voisins dans les embeddings de notre glossaire, que nous ajoutons ensuite dans une section du prompt. Nous utilisons pour ce faire la librairie FAISS, qui est écrite en C++ et propose donc de bien meilleures performances.

On pourrait s'interroger sur la pertinence de comparer les embeddings d'une phrase et d'un mot. Toutefois :

— Il arrive souvent que l'on ne traduise pas une phrase, mais un ou deux mots-clés (c'est très souvent le cas lorsque l'on rencontre un tableau). De même, le glossaire contient des traductions de suites de mots qui forment des expressions types. On n'est donc pas nécessairement dans le cas de la recherche d'un mot dans une phrase, et le calcul de similarité par embedding fait sens.
— En pratique, cette technique se montre satisfaisante et améliore bien la traduction des termes techniques.

6 OCR

Dans l'optique d'un agent multimodal, une des demandes du cahier des charges était de pouvoir traiter les images contenues dans les fichiers, par exemple pour pouvoir traduire les légendes sur un graphique. Nous nous sommes heurtés à de nombreuses difficultés techniques pour implémenter cette fonctionnalité. Notre première approche consistait à :

1. Utiliser pytesseract.image_to_data pour localiser précisément les mots sur le PDF
2. Passer les variables textuelles à l'agent de traduction

Le problème qui s'est présenté est que cela ne fournit que des mots et qu'il faudrait être en capacité de reformer les phrases pour pouvoir traduire correctement (et pas littéralement mot à mot), ce qui n'était pas faisable du fait de la structure parfois compliquée des documents.

Nous avons donc conçu un prototype imparfait, mais qui sert de preuve de faisabilité. Cette approche plus simple consiste à recréer un PDF qui a subi un traitement OCR (en passant par une conversion en image et pytesseract.image_to_pdf_or_hocr), puis à le passer à la pipeline de traduction habituelle. Puisque le texte des images a été reconnu par OCR, il est traité comme le reste des paragraphes. Cependant, il reste des problèmes à la régénération du fichier puisque l'on perd une partie de l'information du PDF. Nous montrons nos résultats dans la présentation.

7 Évaluation de la traduction

Nos encadrants nous ont suggéré de trouver un outil pour mesurer et quantifier la qualité de la traduction de notre programme, le but étant de pouvoir comparer objectivement deux versions différentes de notre programme et de noter nos progrès.

Monsieur Jacquet nous avait mis sur la piste du modèle BLEU [4]. Cependant, nous avons trouvé des modèles plus récents et facilement utilisables, comme le modèle COMET, qui offre une meilleure précision [5]. Voici les résultats obtenus par exemple en comparant notre approche avec et sans RAG :

Fichier	sans RAG	RAG
Rapport d'audit technique	0.8478	0.8485
Rapport de vérification	0.7951	0.7967

Malgré une amélioration tangible de la traduction des termes techniques, nous remarquons que la différence de score est pratiquement nulle. Nous proposons deux explications :

1. Le changement de la traduction des termes techniques ne modifie pas le sens de la phrase, et le modèle est confronté aux mêmes limites que le LLM, ce qui fait que la différence de traduction ne modifie pas le score.
2. Sur un document de 5 pages, le RAG modifie probablement la traduction de quelques termes techniques, ce qui améliore donc le score COMET de la traduction de la phrase. Mais puisque le score global est moyenné sur tout le fichier et qu'il n'y a pas forcément beaucoup de termes techniques, la différence globale de score est très faible.

Nous suggérons donc, pour la suite, de calculer le score de traduction de chaque phrase et de tracer un histogramme des scores obtenus avec et sans RAG pour voir si le RAG permet d'améliorer significativement certaines phrases bien précises (celles contenant des termes techniques).

8 Perspectives d'amélioration

Voici différentes pistes d'amélioration qui nous semblent pertinentes :

— Tester d'autres modèles que celui auquel nous avions accès
— Mettre en place les deux solutions que nous n'avons pas mises en place pour améliorer la traduction, à savoir le fine-tuning et le RAG de phrases traduites pertinentes. Cela nécessitera plus d'exemples de fichiers traduits par des experts
— Paralléliser la pipeline de traduction, actuellement séquentielle
— Réussir à exploiter le modèle d'évaluation de la traduction pour mesurer le succès de chaque version du programme
— Corriger la génération du fichier après traitement OCR

Références

[1] DeepL. *How does DeepL work ?* Consulté le 1 avril 2025. 2021. URL : https://www.deepl.com/fr/blog/how-does-deepl-work.

[2] Longyue Wang et al. *Document-Level Machine Translation with Large Language Models.* 2023. arXiv : 2304.02210 [cs.CL]. URL : https://arxiv.org/abs/2304.02210.

[3] Jiawei Zheng et al. *Fine-tuning Large Language Models for Domain-specific Machine Translation.* 2024. arXiv : 2402.15061 [cs.CL]. URL : https://arxiv.org/abs/2402.15061.

[4] Kishore Papineni et al. “BLEU : a method for automatic evaluation of machine translation”. In : *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL ’02. Philadelphia, Pennsylvania : Association for Computational Linguistics, 2002, p. 311-318. DOI : 10.3115/1073083.1073135. URL : https://doi.org/10.3115/1073083.1073135.

[5] Ricardo Rei et al. *COMET : A Neural Framework for MT Evaluation*. 2020. arXiv : 2009.09025 [cs.CL]. URL : https://arxiv.org/abs/2009.09025.