



ksilink
patient-based drug discovery

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

Développement d'un Large Language Model au service de l'industrie pharmaceutique

Rapport de stage ST40 – P2024

MORALES Léon

Ingénieur
FISE-INFO

KSILINK
16 rue d'Ankara
67 000 Strasbourg
www.ksilink.com

Tuteur en entreprise
OGIER Arnaud

Suiveur UTBM
LOMBARD Alexandre

Remerciements

Je tiens tout d'abord à exprimer ma profonde gratitude à Arnaud Ogier, mon tuteur de stage chez Ksilink. Dès ma candidature spontanée via LinkedIn, il a su réagir avec une rapidité et une disponibilité impressionnante, me répondant et organisant un entretien dans la journée. Tout au long de mon stage, ses conseils avisés, sa patience et sa pédagogie m'ont non seulement permis de progresser mais également de me sentir rapidement à l'aise au sein de l'équipe.

Je remercie également Keyhan et Zahra, qui ont toujours accueilli mes innombrables questions avec bienveillance et enthousiasme. Leur capacité à y répondre avec clarté, ou à me guider vers des solutions adaptées, a été un soutien inestimable dans l'avancement de mes projets.

Mes remerciements s'adressent aussi à l'ensemble des équipes de Ksilink, les "Ksilinkers", qui m'ont accueilli chaleureusement dès le premier jour. Leur convivialité et leur esprit d'équipe ont fait de cette expérience une période mémorable, que ce soit lors des pauses café, des repas de midi ou encore des afterworks.

Enfin, je tiens à remercier Alexandre Lombard, mon référent au sein de l'Université de Technologie de Belfort-Montbéliard. Son suivi attentif de mon stage et ses précieux conseils pour la rédaction de ce rapport m'ont été d'une grande aide tout au long de cette aventure.

Sommaire

Remerciements	1
Table des abréviations :.....	3
I. Introduction	1
A. Ksilink : Une révolution dans la découverte de médicaments	1
B. Le service AI, Image & Data Mining.....	3
C. Mon rôle dans le service	3
II. Travail réalisé.....	4
A. Sujet défini	4
B. Sujet réel	4
C. Antécédents	5
D. Objectifs visés	6
E. Planning daté du travail	7
a) Planning prévisionnel.....	7
b) Planning réel	7
III. Déroulement du travail.....	8
A. Mise à niveau	8
a) Reconnaissance de chiffres manuscrits.....	9
b) Classification d'images de molécules.....	9
B. Projet Cell Morphology-Guided Small Molecule Generation with GFlowNets	19
a) Origine du projet.....	19
b) Objectif du projet.....	19
c) Méthodologie	20
d) Résultats et conclusions	24
e) Apprentissages personnels	24
C. Graphe de connaissances.....	25
a) Objectifs du projet	25
b) Origine du projet	25
c) Ajouts dans le graphe de connaissances	25
d) Améliorations du chatbot	27
e) Tests à l'aide des biologistes	34
f) Rectifications et améliorations	36
g) Conclusion du projet.....	38
IV. Conclusion	40
A. Synthèse des résultats	40
B. Observations	40
C. Estimation du gain financier apporté par mon activité.....	41
V. Bibliographie.....	42
VI. Table des figures.....	45
VII. Annexes	47

Table des abréviations :

- ◊ **CNN : Convolution Neural Network**
Réseau de neurones convolutifs, utilisé principalement pour le traitement et l'analyse d'images.
- ◊ **KG : Knowledge Graph**
Graphe de connaissances, une structure qui relie des entités (objets, concepts) par des relations, utilisée pour représenter et interroger des données complexes.
- ◊ **LLM : Large Language Model**
Catégorie de modèles entraînés à l'aide d'immenses quantité de données pour comprendre et générer des textes en langage naturel. Les exemples les plus connus sont ChatGPT d'OpenAI ou Gemini de Google.
- ◊ **SMILES : Simplified Molecular Input Line Entry System**
Format textuel permettant de représenter la structure chimique des molécules sous forme de chaînes de caractères.
- ◊ **SQL : Structured Query Language**
Langage standard utilisé pour interagir avec des bases de données relationnelles.
- ◊ **CRISPR : Clustered Regularly Interspaced Short Palindromic Repeats**
Technologie de modification génétique permettant de modifier précisément l'ADN.
- ◊ **UMAP : Uniform Manifold Approximation and Projection**
Algorithme de réduction de dimension utilisé pour visualiser des données complexes tout en conservant leurs structures globales.
- ◊ **Graph RAG : Retrieval-Augmented Generation Graph**
Une technique combinant des graphes de données avec des modèles génératifs, utilisée pour améliorer les capacités de recherche et de réponse des modèles.
- ◊ **IA : Intelligence Artificielle**
Discipline informatique visant à concevoir des systèmes capables d'exécuter des tâches nécessitant habituellement l'intelligence humaine, comme la reconnaissance visuelle, le traitement du langage ou la prise de décision.
- ◊ **MNIST : Modified National Institute of Standards and Technology**
Base de données utilisée pour l'entraînement et la validation des algorithmes de classification, composée d'images de chiffres manuscrits.

I. Introduction

A. Ksilink : Une révolution dans la découverte de médicaments

Ksilink[1], fondée en 2014 à Strasbourg, est une association franco-allemande novatrice qui révolutionne la découverte de médicaments grâce à des approches basées sur des modèles cellulaires issus de patients. Avec un statut de PME, cette biotech unique conjugue des compétences exceptionnelles en criblage phénotypique à haut débit (HTS-HCS) et en analyse d'images assistée par l'intelligence artificielle pour développer plus rapidement des thérapies de nouvelle génération.

L'approche de Ksilink repose sur l'utilisation des modèles cellulaires humains reflétant fidèlement les maladies, réduisant ainsi les risques d'échec en phase clinique. En collaborant avec des partenaires académiques, industriels et cliniques, l'organisation combine expertises et ressources pour répondre à des besoins médicaux non satisfaits dans des domaines tels que les maladies neurologiques, musculaires, cardiaques[2] et oncologiques.

Grâce à son réseau d'excellence, Ksilink intègre des capacités uniques de modélisation des maladies, de criblage phénotypique et de développement préclinique. Cette stratégie, qui allie innovation technologique et collaboration interdisciplinaire, permet d'accélérer la mise sur le marché de nouvelles thérapies tout en minimisant les risques.

Ksilink conduit des programmes de découverte de médicaments basés sur des modèles cellulaires humains, ciblant principalement les maladies neurologiques et musculaires. Ces programmes collaboratifs sont développés jusqu'à l'obtention de candidats thérapeutiques innovants, qui sont ensuite transmis à l'industrie pharmaceutique pour leur développement final.

Le lancement de Ksilink a été soutenu par un financement initial de 16 millions d'euros provenant du Programme d'Investissement d'Avenir (PIA), avec le soutien de l'Inserm et de Sanofi. Après une phase de structuration et d'accélération, Ksilink a renforcé son indépendance en recrutant son propre personnel et en investissant dans une plateforme de criblage automatisée. En 2022, ses activités technologiques ont été transférées dans une filiale sous forme de société par actions simplifiée (SAS) afin de valoriser les résultats et de développer des solutions concrètes pour l'industrie.

Ksilink comporte à ce jour 24 membres répartis dans plusieurs services, en voici un aperçu dans la figure 1 :

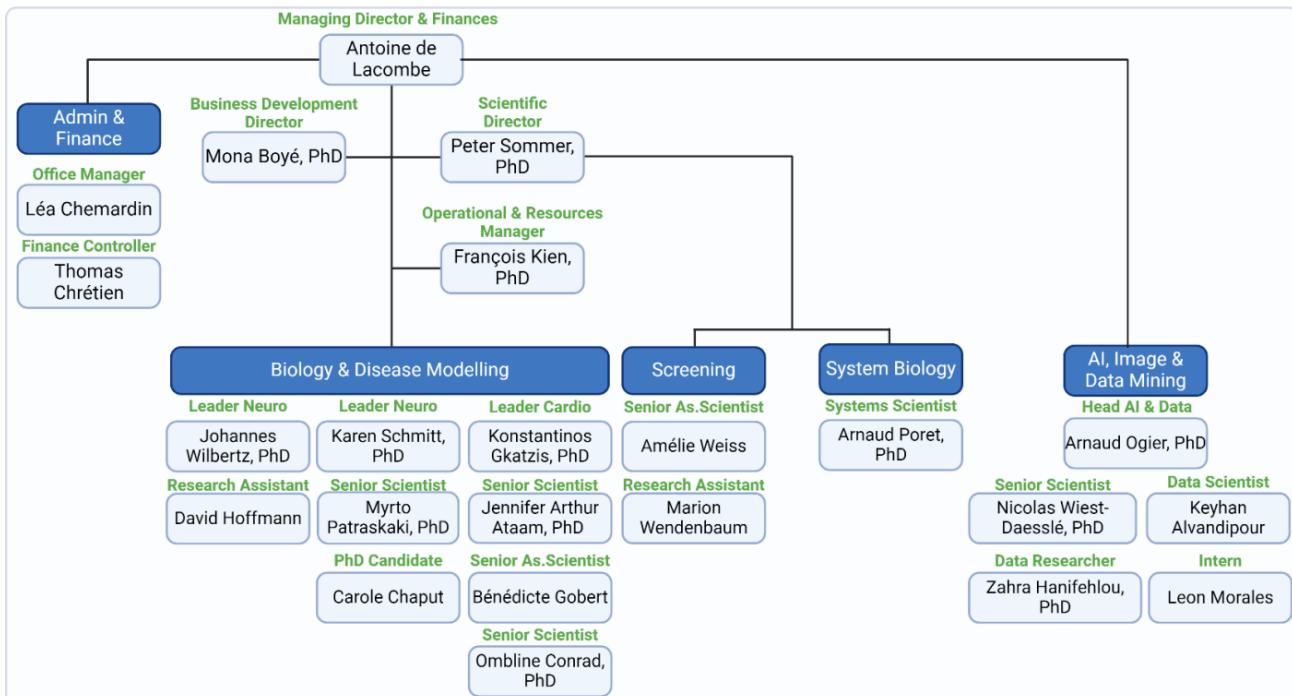


Figure 1: Organigramme de Ksilink

Les principaux services de Ksilink sont :

- ◊ **Biology & Disease Modelling**

Cette équipe se concentre sur la modélisation des maladies humaines grâce à des modèles cellulaires issus de patients. Elle travaille sur des domaines spécifiques comme les maladies neurologiques et musculaires, afin de développer des bases solides pour la découverte de médicaments.
- ◊ **Screening**

Ce département effectue des criblages phénotypiques à haut débit (HTS-HCS) pour identifier des composés prometteurs. L'équipe collabore étroitement avec les autres services pour garantir la pertinence des analyses.
- ◊ **System Biology**

Ce service intègre les données biologiques pour comprendre les interactions complexes entre les systèmes biologiques. Il appuie les projets en fournissant une vision intégrée et en identifiant des cibles thérapeutiques potentielles.
- ◊ **AI, Image & Data Mining (Service auquel j'appartiens)**

Ce département exploite l'intelligence artificielle et l'analyse avancée d'images pour extraire des informations précieuses à partir des modèles cellulaires.

B. Le service AI, Image & Data Mining

Le service AI, Image & Data Mining est un pilier central de l'innovation chez Ksilink. Il exploite des technologies de pointe en intelligence artificielle pour optimiser le criblage phénotypique à haut débit et ainsi accélérer le développement de nouvelles thérapies. En analysant des données massives issues d'images cellulaires complexes, ce service identifie les molécules prometteuses capables de corriger des anomalies associées à des maladies graves.

Grâce à des algorithmes avancés, l'IA permet une analyse fine de milliers de paramètres cellulaires, différenciant avec précision les phénotypes sains et malades. Le service joue un rôle clé dans le traitement des images générées par le criblage automatisé, en utilisant des techniques d'apprentissage profond pour détecter des changements subtils dans les cellules exposées à des substances actives. Cette approche dérisque la recherche en identifiant tôt les candidats-médicaments les plus prometteurs, tout en réduisant les coûts et délais du développement préclinique.

En combinant innovation technologique et expertise scientifique, le service AI, Image & Data Mining participe à la mission de Ksilink : transformer la recherche translationnelle en solutions thérapeutiques tangibles.

C. Mon rôle dans le service

Durant mon stage au sein du service Ai, Image & Data Mining, j'ai eu l'opportunité de contribuer à deux projets étroitement liés à l'intelligence artificielle :

Un projet novateur basé sur des recherches récentes

Le premier projet, entièrement nouveau pour Ksilink, s'appuyait sur un article scientifique[3] publié en août 2024. En tant que principal contributeur, j'ai pu développer ce projet en autonomie, tout en bénéficiant du soutien précieux de mon tuteur de stage, Arnaud OGIER, et de Keyhan ALVANDIPOUR, qui supervisait l'avancement.

Complétion d'un graphe de connaissances et amélioration d'un chatbot existant

Le second projet portait principalement l'ajout d'informations au graphe de connaissances et sur l'amélioration d'un chatbot, préalablement implémenté par Zahra et Arnaud. Mon rôle a consisté à optimiser les fonctionnalités et la performance de cet outil, en m'appuyant sur l'expertise de mes collègues pour intégrer de nouvelles solutions adaptées aux besoins du service et rendre son utilisation accessible et utile aux scientifiques des autres services.

Ces deux projets m'ont offert une immersion complète dans l'écosystème de Ksilink, me permettant d'appliquer mes connaissances à des problématiques réelles tout en contribuant à des avancées concrètes dans la recherche médicale.

II. Travail réalisé

A. Sujet défini

Le sujet de mon stage portait sur le développement d'un **Large Language Model (LLM)** appliqué à l'industrie pharmaceutique, en exploitant les données internes de Ksilink. L'objectif principal était de créer une interface combinant un **graphe de connaissances (Knowledge Graph, KG)** et un LLM, afin de mieux exploiter les relations entre différents concepts extraits de données variées tout en gardant les données en interne.

Pour atteindre cet objectif, le projet reposait sur l'utilisation de frameworks open source tels que **OLLAMA** ou des alternatives similaires, intégrés avec des outils comme **PyTorch** pour la conception et l'entraînement des modèles. Cette approche visait à structurer les données complexes des bases de Ksilink tout en les rendant plus accessibles. Grâce à cette solution, les biologistes pourraient explorer ces informations de manière intuitive via une interface en langage naturel.

B. Sujet réel

Au cours du stage, le projet a évolué pour répondre à des défis techniques spécifiques, en particulier liés à l'intégration de données hétérogènes. Le sujet a été divisé en deux volets distincts :

1. **Construction d'un espace latent commun** : Cette étape visait à intégrer plusieurs types de données, notamment des descriptions moléculaires (en format SMILES) et des informations extraites d'images de cellules. L'objectif était de représenter ces données dans un espace vectoriel partagé, facilitant ainsi leur analyse et leur exploitation conjointe, dans l'optique de créer un générateur de nouvelles molécules.
2. **Graphe de connaissances et lien avec le LLM** : Cette partie avait pour but d'enrichir le graphe de connaissances en ajoutant de nouvelles relations et informations. Cela incluait également l'amélioration de l'interface reliant le graphe de connaissances à un LLM, pour permettre des interactions en langage naturel et l'ajout de fonctionnalité rendant l'outil plus instinctif et compréhensible.

C. Antécédents

Pour me préparer à ce stage et acquérir une première base de connaissances en intelligence artificielle (IA), j'ai entrepris plusieurs démarches personnelles avant mon arrivé chez Ksilink.

Tout d'abord, j'ai étudié l'histoire et les évolutions de l'IA à travers des ouvrages spécialisés [4,5]. Cela m'a permis de comprendre les fondements théoriques et les progrès majeurs qui ont marqué ce domaine en constante évolution. En complément, cette exploration m'a donné une vision globale des champs d'application et des défis liés à l'intelligence artificielle.

Ensuite, j'ai développé plusieurs projets personnels visant à mettre en pratique des concepts de base en programmation et en IA. Parmi ces projets, j'ai réalisé :

- ◊ Un jeu **Snake automatique**, où j'ai implémenté un algorithme pour que le serpent trouve automatiquement sa route vers la nourriture.
- ◊ Une version avancée du jeu **Morpion**, incluant une variante appelée **Ultimate Morpion**, composée de 9 grilles connectées où le but est de gagner trois grilles alignées.

Ces expériences m'ont permis de consolider mes compétences en algorithmique.

Cependant, malgré ses efforts, j'ai rapidement constaté que ces projets ne suffisaient pas à répondre au niveau d'exigence de mon stage. Les problématiques rencontrées chez Ksilink, notamment en reconnaissance d'images et en apprentissage profond, demandaient des compétences bien plus spécifiques. Pour combler ce fossé, les premières semaines de mon stage ont été consacrées à une montée en compétences intensives. Cela a inclus une formation sur des concepts avancés tels que :

- ◊ Les réseaux de neurones convolutifs (CNN) pour le traitement d'images.[6,7]
- ◊ Les techniques de classification[8] et de clustérisations pour organiser et analyser les données.
- ◊ L'utilisation d'outils de visualisation et d'évaluation des modèles.[9]

Ces efforts d'apprentissage m'ont permis d'acquérir les bases nécessaires pour relever les défis du projet, tout en me permettant d'intégrer efficacement les méthodologies employées par Ksilink.

D. Objectifs visés

Les objectifs de mon stage étaient organisés en plusieurs étapes, avec un équilibre entre l'acquisition de compétences techniques et leur application pratique dans le cadre des projets de Ksilink.

1. Acquisition de compétences en intelligence artificielle :

- ◊ Développer une compréhension approfondie des bases théoriques de l'intelligence artificielle, en particulier des algorithmes d'apprentissage automatique et des réseaux de neurones.
- ◊ Réaliser des calculs à la main pour les principales étapes des modèles d'apprentissage supervisé, comme la backpropagation, afin de maîtriser les concepts sous-jacents.

2. Première mise en pratique en vue d'une autonomie future :

- ◊ Concevoir un réseau convolutif (CNN) capable de classifier des images prise à l'aide de microscopes en quatre catégories. Ce projet avait pour objectif d'appliquer mes connaissances théoriques à un cas concret tout en utilisant des outils d'analyse pour visualiser les résultats.
- ◊ Explorer des techniques avancées d'interprétation de modèles, comme l'intégration de gradient et la visualisation des couches d'activation, pour mieux comprendre les erreurs et les limites du modèle.

3. Étude scientifique et reproduction des résultats :

- ◊ Étudier en détail l'article scientifique[3] à la base du projet principal, afin d'en comprendre les méthodologies employées.
- ◊ Reproduire les résultats décrits dans l'article avec les données internes à Ksilink, notamment la création d'un espace latent commun pour intégrer des données hétérogènes, et explorer les applications possibles à partir de cet espace.

4. Enrichissement du graphe de connaissances de Ksilink :

- ◊ Ajouter des informations pertinentes et établir de nouvelles relations au sein du graphe de connaissances existant, qui comprend plus de 100 000 nœuds et 2,5 millions de relations.
- ◊ Tester de multiples approches pour exploiter le KG :
 - ◆ Implémenter un chatbot capable de répondre à des requêtes en langage naturel grâce au graphe de connaissances de Ksilink.
 - ◆ Implémenter de nouvelles fonctions au chatbot (visualisation des réponses sous forme de graphique)
 - ◆ Entraîner un modèle pour générer de nouvelles connexions entre les nœuds.

Ces objectifs reflètent une progression logique, allant de la formation théorique à des projets concrets, tout en intégrant des méthodologies avancées et des outils adaptés. Ils ont guidé mon travail tout au long du stage, tout en me permettant d'approfondir ma compréhension des technologies utilisées chez Ksilink.

E. Planning daté du travail

a) Planning prévisionnel

Le planning initial de mon stage était organisé en plusieurs phases principales, chacune correspondant à une étape clé dans l'atteinte des objectifs fixés :

1. **Étape 1 : 1 mois** : Formation et montée en compétences.
 - ◊ Comprendre les bases théoriques et pratiques de l'intelligence artificielle, notamment le traitement d'images et les méthodes d'apprentissage profond comme les réseaux de neurones convolutifs (CNN).
 - ◊ Réaliser des projets introductifs pour maîtriser les outils et concepts nécessaires au projet principal.
2. **Étape 2 : 2 mois** : Développement du premier projet.
 - ◊ Reproduire les résultats d'un article scientifique en construisant un espace latent commun pour intégrer des données hétérogènes (structures moléculaires et données d'imagerie).
 - ◊ Atteindre un premier résultat concret avant la fin de cette phase.
3. **Étapes 3 : 1 mois** : Approfondissement du projet principal
 - ◊ Générer des données à partir de l'espace latent commun.
4. **Étape 4 : Jusqu'à la fin du stage** : Intégration au graphe de connaissances.
 - ◊ Ajouter de nouvelles relations et informations au graphe de connaissances.
 - ◊ Implémenter un chatbot permettant des requêtes en langage naturel.
 - ◊ Entraîner un modèle capable de prédire de nouvelles relations entre les nœuds si possible.

b) Planning réel

En raison des défis techniques rencontrés, notamment liés à la qualité et à l'hétérogénéité des données, le planning a été ajusté comme suit :

1. **Étape 1 : Formation initiale (Semaines 1 à 4)**
 - ◊ Création d'un réseau convolutif simple (CNN) pour reconnaître des chiffres manuscrits (jeu de données MNIST).
 - ◊ Développement d'un CNN plus complexe pour classifier des images de molécules issues de Ksilink, avec intégration d'outils de visualisation comme TensorBoard.
 - ◊ Ces étapes ont respecté le planning initial et m'ont permis de maîtriser les bases nécessaires pour la suite.
2. **Étape 2 : Reproduction des résultats scientifiques (Semaines 5 à 13)**
 - ◊ Tentative de reproduire les résultats décrits dans l'article avec les données de Ksilink.

- ◊ Bien que les résultats aient été satisfaisants avec les données de l'article, il a été impossible de les reproduire avec les données internes en raison de différences significatives dans leur structure et leur qualité.
- ◊ Cette difficulté a conduit à la mise en pause de ce projet.

3. Étape 3 : Abandon de la génération à partir de l'espace latent

- ◊ En raison des limitations rencontrées à l'étape précédente, cette phase n'a pas pu être réalisée et a été reportée en attente de données plus adaptées.

4. Étape 4 : Enrichissement du graphe de connaissances (Semaines 14 à fin du stage)

- ◊ Ajout de nouvelles données au graphe de connaissances pour améliorer sa richesse et sa pertinence.
- ◊ Préparation d'un environnement d'entraînement avec un petit graphe expérimental pour éviter tout impact négatif sur le graphe principal.
- ◊ Intégration du chatbot lié le graphe de connaissances

Malgré les ajustements nécessaires, le planning réel m'a permis de respecter une progression logique et d'atteindre des résultats concrets dans des parties importants du projet. Les imprévus liés aux données ont été une occasion d'apprendre à adapter mes approches et de prioriser les tâches en fonction des contraintes rencontrées.

III. Déroulement du travail

A. Mise à niveau

La phase de mise à niveau s'est déroulée sous la forme de deux projets réalisés en grande partie de manière autonome, avec le soutien ponctuel de Keyhan ALVANDIPOUR et Arnaud OGIER pour répondre à mes questions et me guider. Cette étape a été essentielle pour acquérir les compétences et connaissances nécessaires au déroulement du reste de mon stage. J'ai de plus effectué un rapide exercice pour me familiariser à la manipulation des fichiers .tar ([Annexe 2 et 3](#))

Comprendre les bases : perceptron et backpropagation

Avant de débuter les projets, j'ai pris le temps de me familiariser avec des concepts fondamentaux du machine learning. Par exemple, j'ai étudié le fonctionnement d'un perceptron, une unité de base des réseaux neuronaux, pour mieux comprendre sa logique. J'ai aussi effectué manuellement (Figure 2) les calculs de backpropagation[10], une méthode permettant d'ajuster les poids d'un réseau neuronal afin de minimiser

l'erreur entre ses prédictions et les valeurs attendues. Ces calculs m'ont permis d'assimiler concrètement les principes mathématiques utilisés.

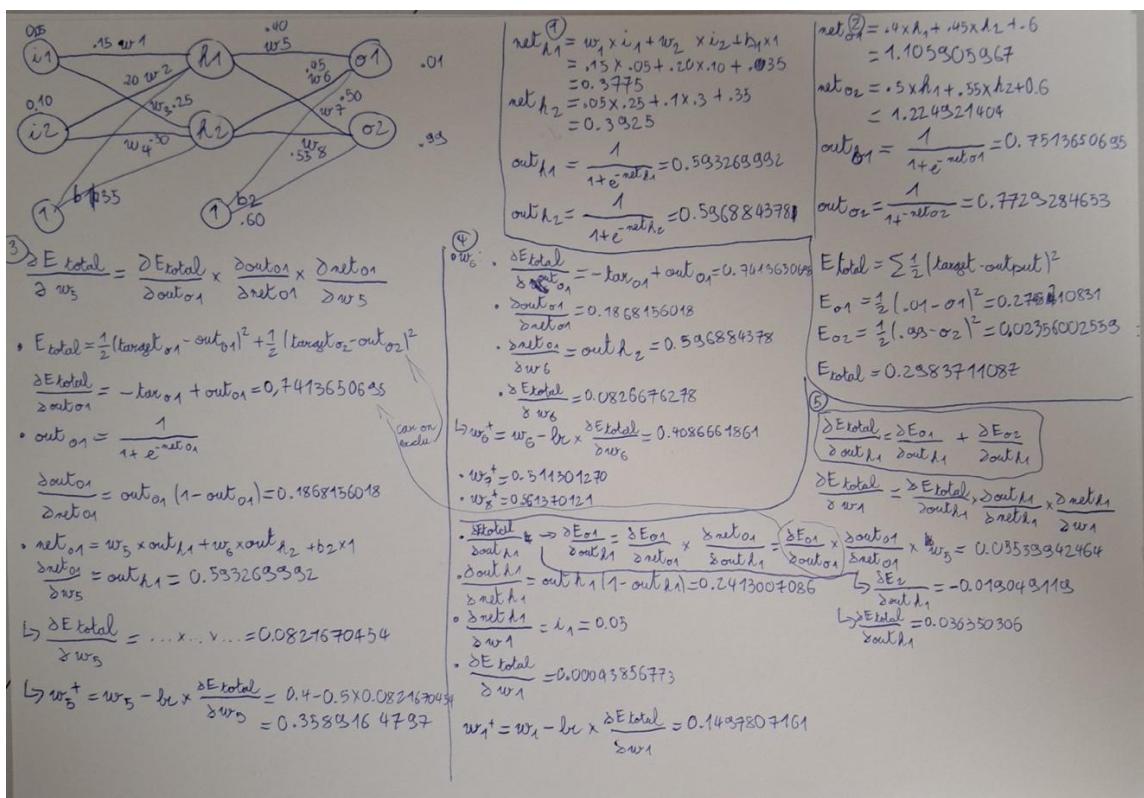


Figure 2: Recto de mon brouillon de calcul de back propagation

a) Reconnaissance de chiffres manuscrits

Pour mon premier projet j'ai développé, en utilisant Python et plus précisément Pytorch, un réseau de neurones simple capable de reconnaître des chiffres manuscrits de 0 à 9. Ce projet, souvent considéré comme un exercice d'introduction en machine learning, m'a permis d'explorer des concepts clés :

- ◊ **Fonctions de perte et d'activation** : elles influencent la manière dont le réseau évalue et ajuste ses performances.
- ◊ **Learning rate** : le paramètre contrôlant la vitesse d'apprentissage du modèle.
- ◊ **Optimiseurs** : algorithmes qui ajustent les poids du réseau de manière efficace.

Grâce aux nombreux tutoriels disponibles en ligne[6,7], j'ai pu construire un modèle initial précis et acquérir une bonne compréhension de ces notions. Ce travail a posé les bases pour le projet suivant, plus complexe.

b) Classification d'images de molécules

Le deuxième projet visait à classer des images de cellules générées chez Ksilink en quatre catégories distinctes selon leurs activités et leurs effets. Dans cette partie, ces différents groupes seront nommés groupe 1, 2, 3, 4 pour ne pas divulguer les différentes

catégories. Cette étape a été particulièrement enrichissante, car elle m'a permis de travailler sur des aspects plus avancés :

- ◊ **Préparation du jeu de données** : Créer un jeu de données s'est révélé crucial pour assurer la fiabilité de l'entraînement du modèle. Une base de données bien structurée réduit les biais et erreurs susceptibles de fausser les résultats.
- ◊ **Optimisation du modèle** : Après avoir conçu un modèle initial, j'ai intégré des techniques comme le **dropout**, des tests sur différents paramètres, et l'analyse avec TensorBoard[11] pour identifier et corriger les problèmes observés.

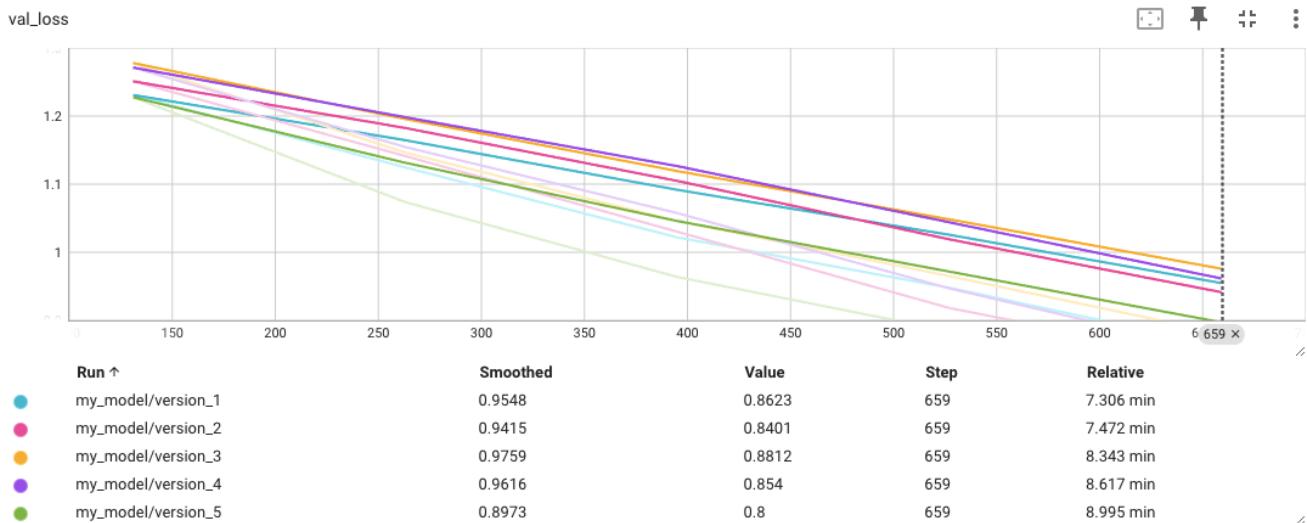


Figure 3: Validation Loss du model dans les premiers essais

La fonction de perte représente l'écart entre la sortie prédite du modèle et la valeur cible réelle. Pendant l'entraînement, le modèle tente de minimiser cette perte.

La **validation loss** (ou perte de validation) mesure les performances du modèle sur les données de validation, c'est-à-dire des données non vues pendant l'entraînement.

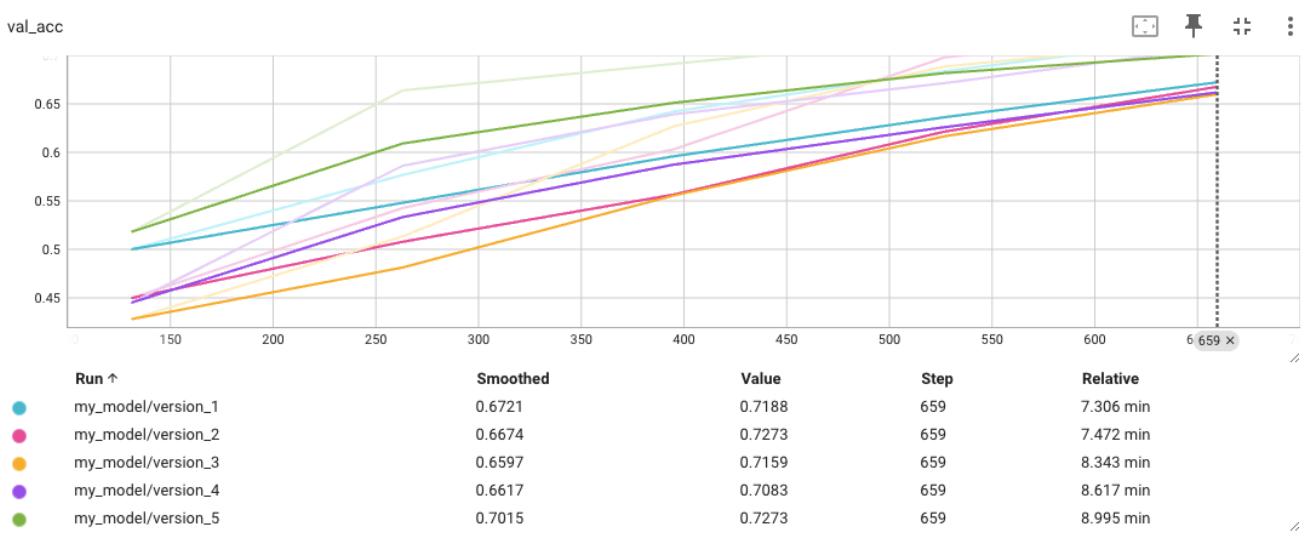


Figure 4: Validation accuracy du model dans les premiers essais

La **validation accuracy** (ou précision de validation) indique la proportion de prédictions correctes que le modèle réalise sur les données de validation.

Sur ces deux graphiques (figure 3 et 4), récupérer grâce à Tensorboard, on peut voir que les courbes de validation loss et de validation accuracy continuent respectivement de monter et de descendre sans se stabiliser, le modèle n'a donc pas encore atteint son point de convergence. Il faut donc augmenter le nombre de répétitions d'entraînement pour laisser au modèle le temps d'apprendre.

Après m'en être rendu compte, voici le rendu :

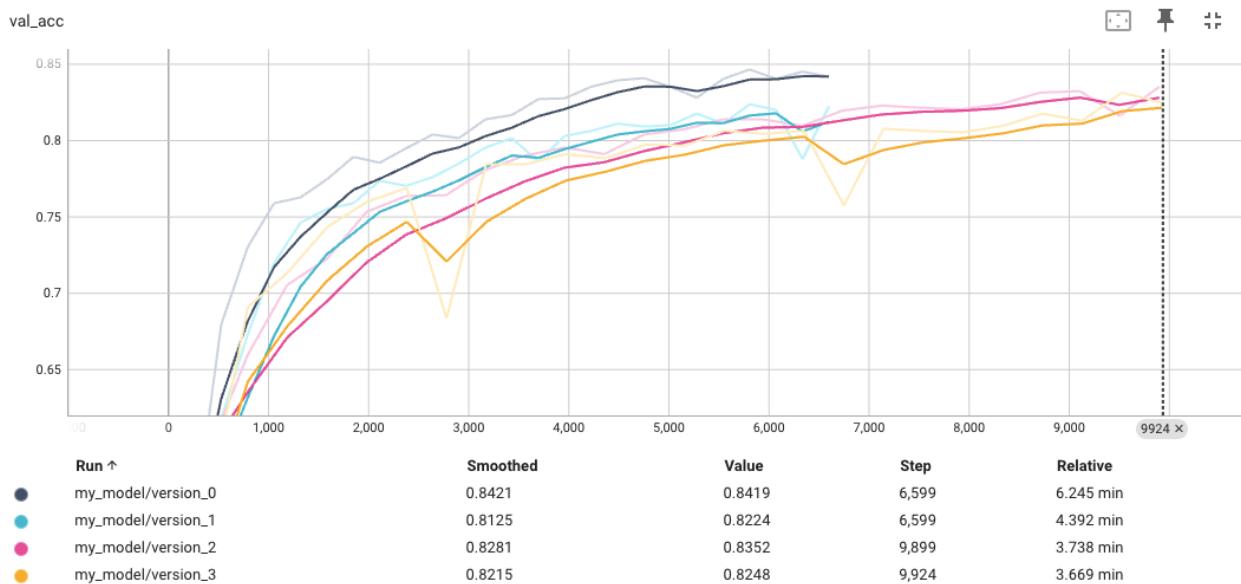


Figure 5: Validation accuracy du model après avoir modifié le nombre de répétitions d'entraînements

Sur la figure 5, on peut voir que la précision de validation du modèle commence à se stabiliser aux alentours de 0.84. En laissant au modèle plus de temps pour s'entraîner, on voit bien que cela lui a permis d'apprendre plus de choses et d'affiner ces prédictions.

Par la suite, je suis passé de **pytorch** à **pytorch lightning**[12], qui sera utilisé dans les projets suivants. En plus de me préparer à la suite de mon stage, cela m'a permis de restructurer mon code et de le rendre plus clair par la même occasion.

Voyant que le modèle ne progressait plus, j'ai cherché les raisons et des nouvelles méthodes pour l'améliorer.

Optimisation avancée : Normalisation et augmentation des données

Pour améliorer les performances du modèle, j'ai exploré plusieurs méthodes avancées. Deux approches ont particulièrement retenu mon attention : la **normalisation** et **l'augmentation des données**.

- ◊ **Normalisation des données** : Cette technique consiste à réduire les écarts significatifs entre les valeurs des données d'entrée, afin d'éviter que certaines valeurs dominent les autres dans le processus d'apprentissage.
- ◊ **Augmentation des données**[13] : L'objectif ici est de rendre le modèle plus robuste et capable de généraliser à de nouvelles données. Pour ce faire, j'ai introduit des modifications sur les images du jeu de données, telles que :
 - ◆ Rotation aléatoire de 90°.

- ◆ Inversion horizontale.
- ◆ Ajout de bruit.
- ◆ Effacement partiel d'une image.

Ces augmentations étaient appliquées à 50 % des images d'entraînement, et chaque image modifiée pouvait comporter une ou plusieurs transformations. J'ai cependant constaté que certaines augmentations n'étaient pas adaptées, et j'ai donc ajusté les paramètres pour ne conserver que celles apportant des améliorations significatives et enlever les autres comme le changement de couleurs qui dans mon cas ne sert à rien.

De plus, pour améliorer la reproductibilité de l'entraînement et limiter les effets du hasard dans la répartition des données (entraînement, validation, test) et dans l'application des augmentations, j'ai fixé une graine aléatoire. Cela m'a permis de mieux analyser les impacts des modifications sur les performances du modèle.

Améliorations du modèle : Réduction et régularisation

Face aux limites rencontrées par le modèle, j'ai entrepris plusieurs ajustements pour améliorer sa performance :

- ◊ **Réduction de la taille du modèle** : Un modèle plus compact apprend plus rapidement et est moins susceptible de surajustement (*overfitting*).
- ◊ **Scheduler pour le learning rate** : Ce mécanisme réduit progressivement la vitesse d'apprentissage au cours des itérations, permettant au modèle de se stabiliser sur une solution optimale.
- ◊ **Weight decay** : Une technique de régularisation visant à contrôler la modification des poids pour éviter que le modèle ne s'ajuste trop aux données d'entraînement.

Ces ajustements ont permis d'obtenir un modèle mieux optimisé (Figure 6), dont les performances étaient plus cohérentes.

Voici le modèle final utilisé (Figure 6) :

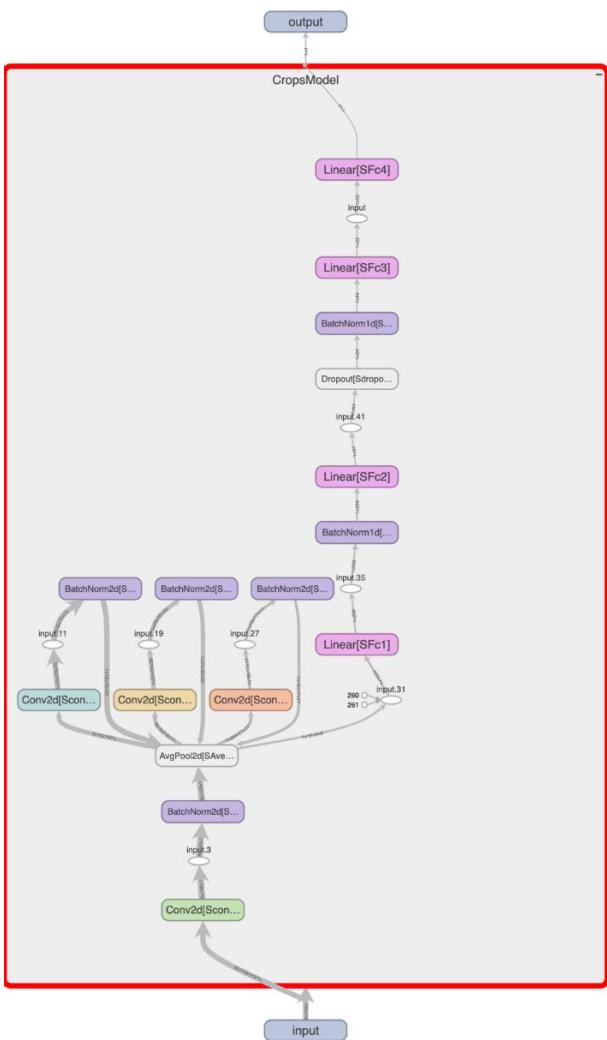


Figure 6: Graphique du modèle via tensorboard ([Annexe 1](#) pour avoir une version pleine page)

Le modèle final présentait une architecture optimisée, prenant en entrée des images de dimensions 140×140 avec 4 canaux, et appliquant successivement des couches convolutives. Chaque couche réduisait la taille de l'image tout en augmentant le nombre de canaux, aboutissant à une sortie de 17×17 pixels avec 24 canaux (voir Annexe 4 et 5). Cette structure a permis de conserver les caractéristiques essentielles des images tout en éliminant les détails superflus.

Analyse des erreurs et visualisation

Pour mieux comprendre les limites du modèle, j'ai utilisé divers outils de visualisation :

- Matrice de confusion** : Cette représentation m'a permis d'identifier les catégories que le modèle confondait fréquemment. Par exemple, les premières matrices ont montré des difficultés à différencier le premier et le deuxième groupe, ainsi que le troisième et le quatrième groupe (Figure 7). Ces observations

ont été confirmées par Keyhan, qui a souligné la similarité visuelle entre ces groupes.



Figure 7: Matrice de confusion à l'étape 1 où chaque colonne du tableau contient une classe prédictive par l'algorithme et les lignes des classes réelles. Les valeurs sur la diagonale représentent les prédictions correctes alors que les valeurs en dehors de la diagonale sont celle incorrectement prédites par le modèle.

2. Images mal classifiées : En affichant les exemples incorrects, j'ai pu analyser les prédictions du modèle et vérifier la validité des labels. Cela m'a conduit à conclure que certaines erreurs provenaient d'une mauvaise labélisation des données.

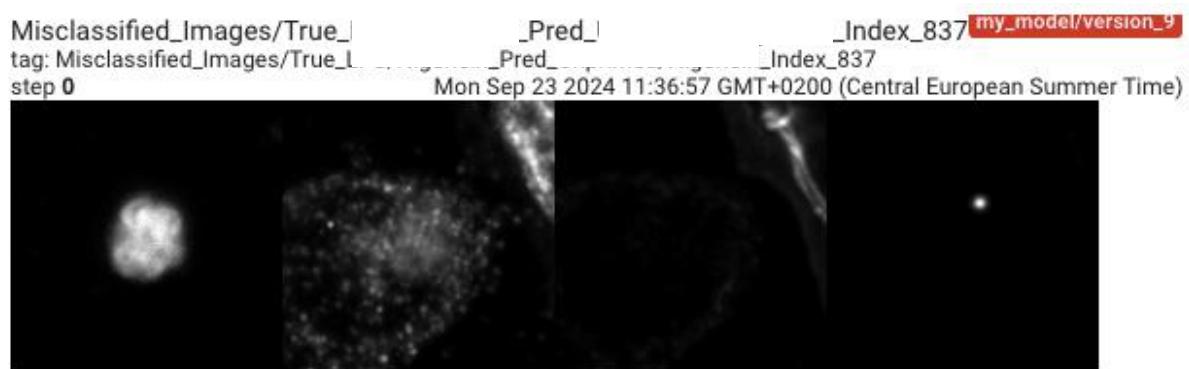


Figure 8: Affichage des quatre canaux d'une image mal classifiée

Chaque image ici représente le même composé dans un canal différent. Ce composé appartient au groupe un alors qu'il est classifié par le modèle comme un groupe deux. Ces images m'ont permis de confirmer ce que m'avait dit Keyhan.

3. **Visualisation des gradients[9]** : En utilisant les descentes de gradients, j'ai observé les zones de l'image que le modèle considérait comme importantes. Par exemple l'analyse de la figure 9 a révélé que le modèle s'appuyait principalement sur un canal spécifique, ce qui a orienté mes efforts pour équilibrer l'utilisation des canaux d'entrée.

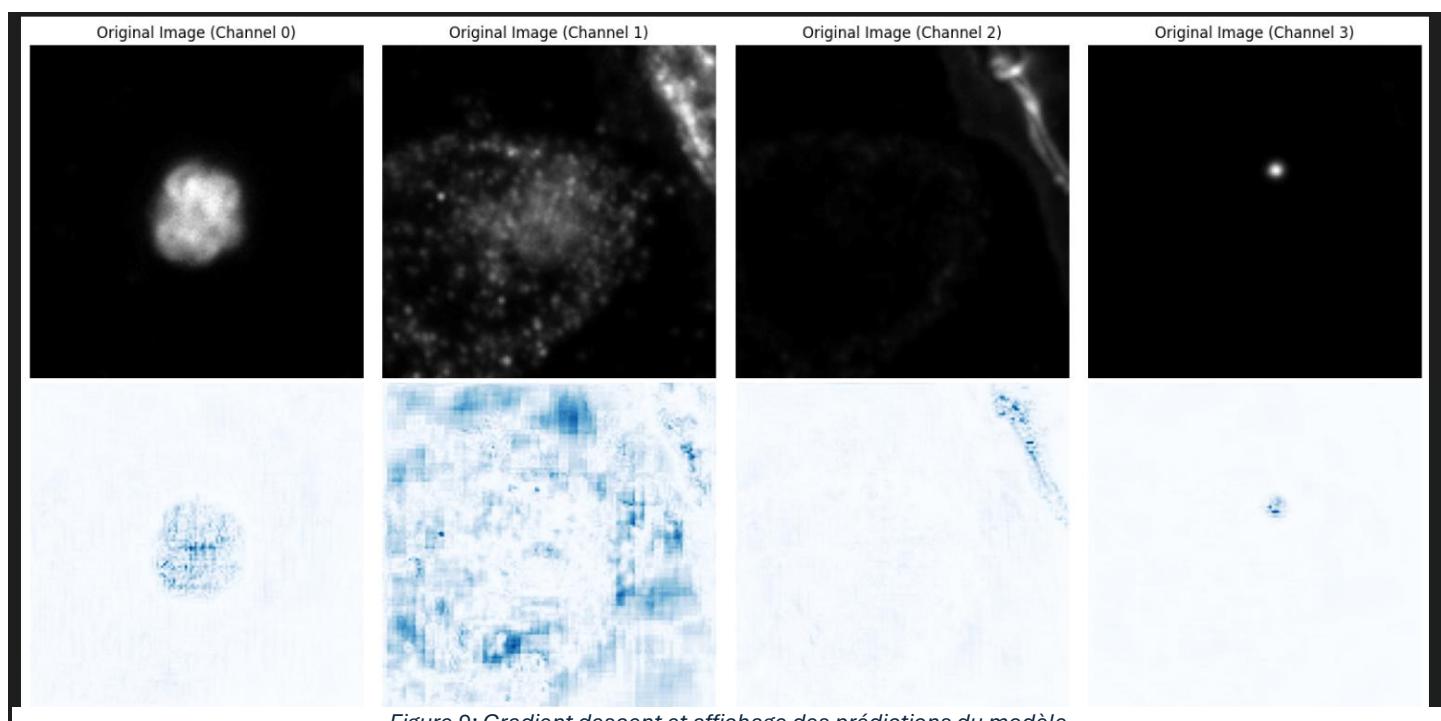


Figure 9: Gradient descent et affichage des prédictions du modèle

Sur ces images, plus le bleu est intense plus le modèle porte d'importance à cette partie. Cela permet de comprendre ce qui est important pour le modèle.

Pour une meilleure visualisation, j'ai décidé d'afficher cela selon la couche convolutive.

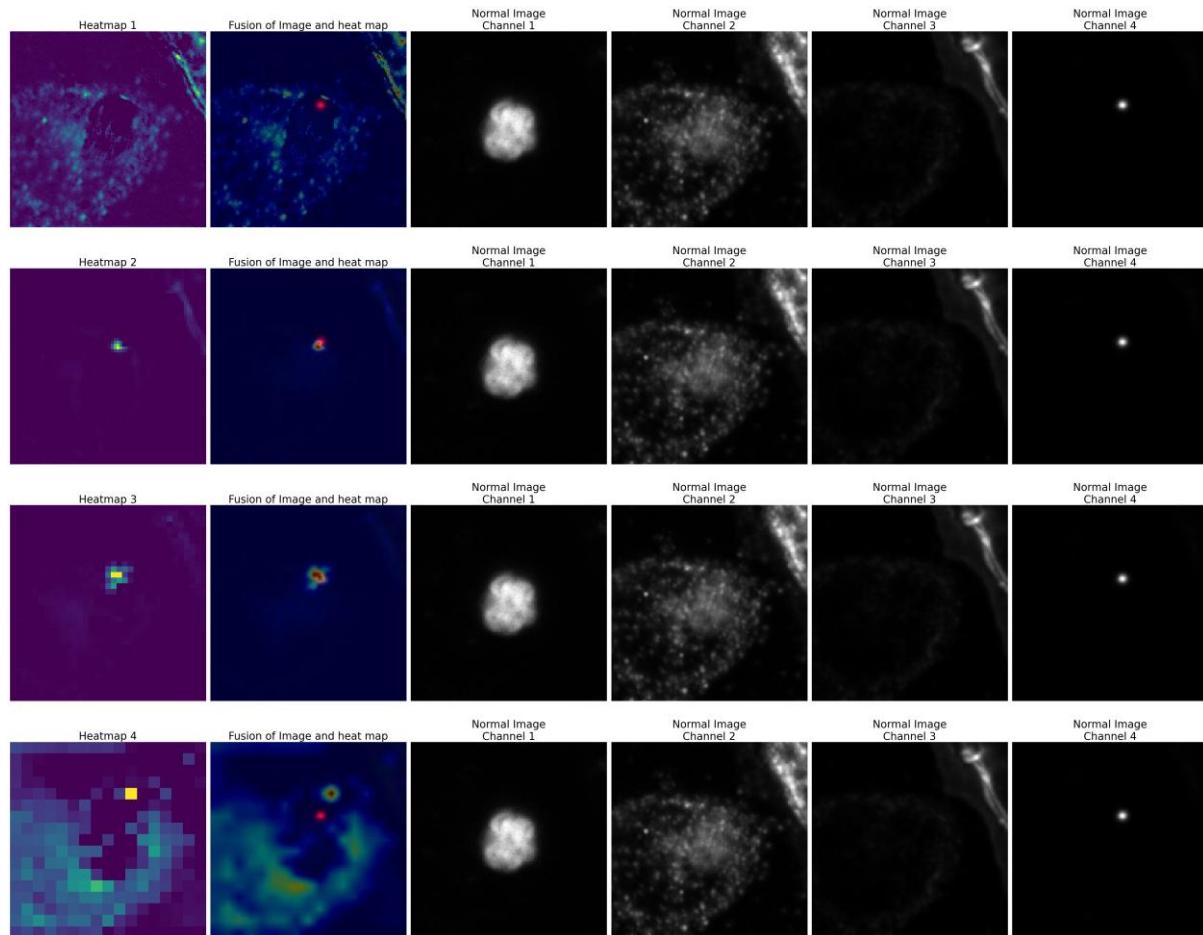


Figure 10: Visualisation selon la couche convective

Cette image (Figure 10) est décomposée en plusieurs parties, d'abord en colonnes. La première colonne montre la heatmap générée à partir des descentes de gradients, c'est la partie la plus importante de l'image, celle qui permet réellement de comprendre ce que voit le modèle, ici plus la couleur tend vers le rouge plus c'est important. La deuxième colonne est une superposition de la heatmap et des images permettant de localiser les informations, mais attention cette colonne peut être biaisée par l'intensité d'une des images superposées, par exemple sur la première ligne on voit un point rouge qui n'apparaît pas sur la heatmap, il s'agit du point de la dernière image, il faut donc faire attention et plutôt se baser sur la heatmap. Les 4 dernières colonnes montrent les 4 canaux d'entrées. Et en ligne, on a les différentes couches convective, en partant de la première avec des images de 128x128 pixels jusqu'à avoir des images sur la dernière ligne de 17x17 pixels.

On peut alors voir que la partie la plus importante de l'image pour le modèle est le point dans le canal 4, c'est ce point qui influence le plus la prédiction du modèle. On peut aussi voir que dans la première couche convective, le modèle ne l'avait pas repéré et qu'il s'affiche de plus en plus fort au fur et à mesure.

[Annexes 4 et 5](#) pour voir les visuels de chaque canal des couches 1 et 4.

Une autre étape importante de l'optimisation a consisté à évaluer la pertinence des canaux d'entrée du modèle. En entraînant le modèle tout en supprimant un ou plusieurs canaux, j'ai pu constater que le troisième canal n'avait qu'un impact négligeable sur les résultats. Cette observation m'a permis de simplifier le traitement des données sans compromettre la précision du modèle.

Une fois tout cela fait, voici mes résultats finaux :

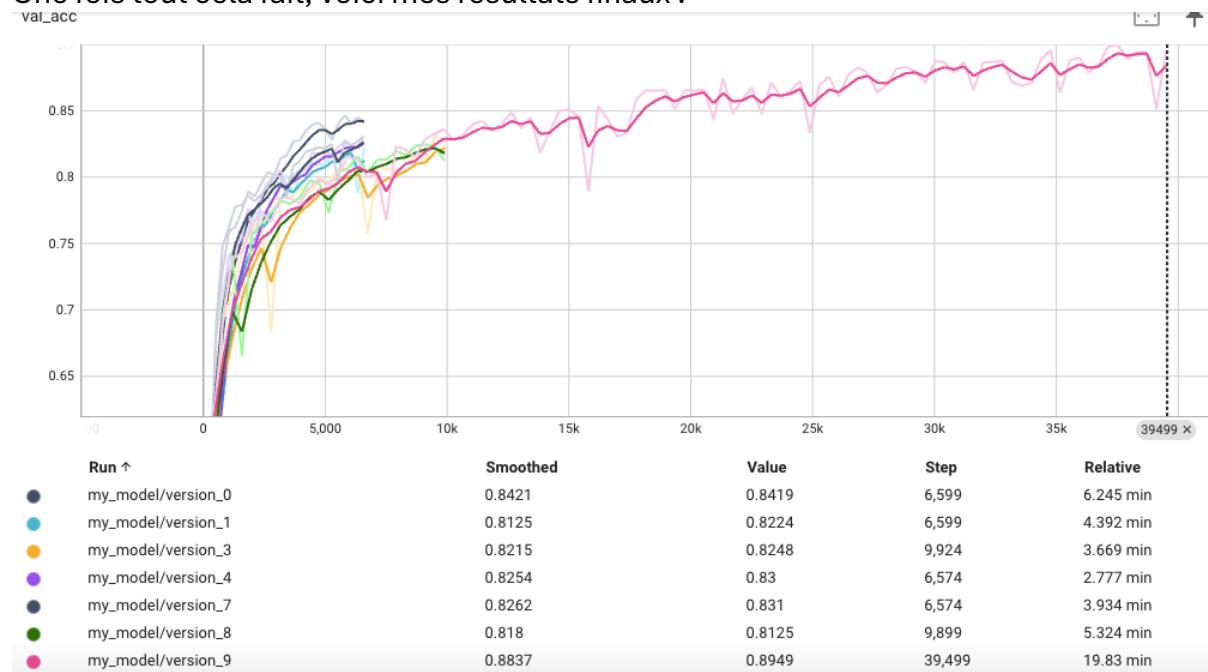


Figure 11 : Validation accuracy final

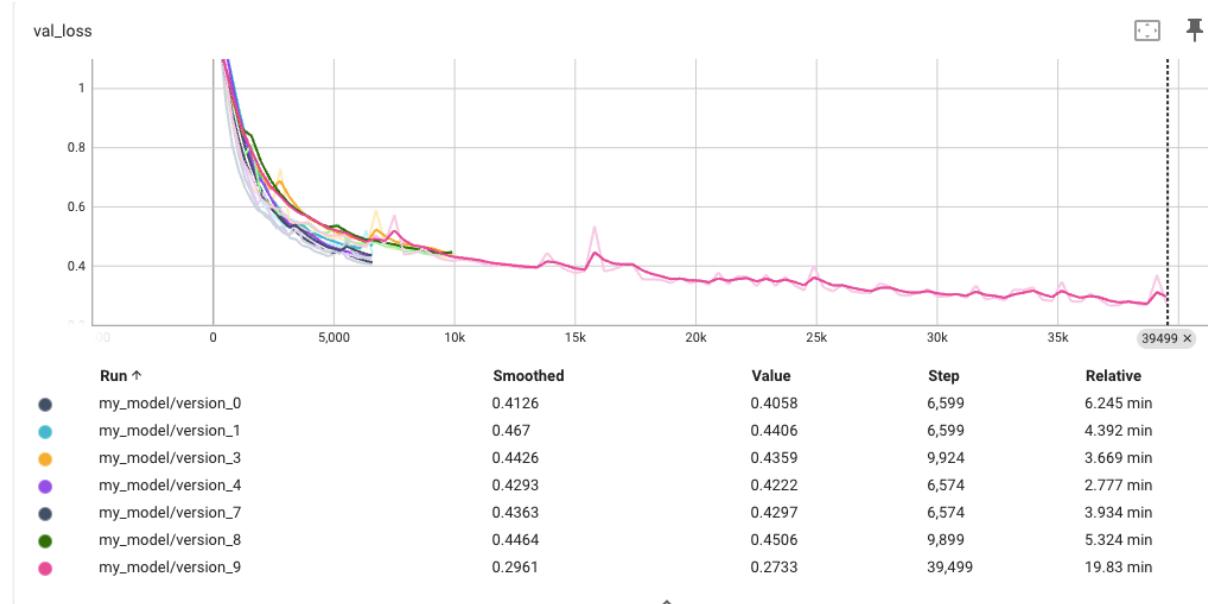


Figure 12: Validation loss final

On peut voir sur les courbes de la figure 11 que le modèle se stabilise aux alentours de 0.87 de précision ce qui est plutôt satisfaisant, continuer l'entraînement risquerai de créer du surapprentissage.

Sur les courbes de la figure 12, on peut remarquer que certaines versions semblent plus précises que la version finale. Cela s'explique par une erreur lors de l'augmentation¹ des images, certaines images augmentées se retrouvaient dans les données de validation. Or, ces images augmentées ne doivent pas fuiter depuis les données d'entraînement, puisqu'elles sont des copies similaires provenant d'autres images. Cette fuite faussait donc les résultats. Après avoir corrigé cette erreur, la version 9 de mon modèle s'est avéré la plus précise.

Les résultats finaux, bien qu'assez satisfaisants, n'étaient pas aussi précis que prévu, principalement en raison des limitations du jeu de données :

- ◊ **Labels incorrects** : Certains échantillons étaient mal labellisés en raison de variations au sein d'une même plaque expérimentale.
- ◊ **Jeu de données limité** : Une base de données plus grande et mieux étiquetée aurait permis au modèle de progresser davantage et de réduire les phénomènes de surajustement.

Malgré les tentatives de palier à ces problèmes, notamment en appliquant des techniques d'augmentation des données, je n'ai pu observer que des améliorations modestes. Par conséquent, un jeu de données plus important et de meilleure qualité resterait la solution la plus efficace pour des résultats plus robustes.

Conclusion de la mise à niveau

Cette phase de mise à niveau m'a permis de :

1. Maîtriser les bases des réseaux de neurones et de leurs principes fondamentaux, comme le perceptron, la backpropagation, et les fonctions d'activation.
2. Comprendre et implémenter des techniques avancées pour l'optimisation des modèles, telles que le dropout, le weight decay, et l'utilisation d'un scheduler.
3. Explorer des outils de visualisation et d'analyse des erreurs, tels que les matrices de confusion et les descentes de gradients.
4. Appréhender l'importance de la préparation et de la qualité des données dans la réussite d'un projet en machine learning.

Ces apprentissages ont constitué une base solide pour la suite de mon stage et m'ont préparé à aborder des problématiques plus complexes avec assurance et méthode et permirent une certaine autonomie pendant la suite de mon stage.

¹ Augmentation : Ensemble de techniques générant des versions modifiées d'images, permettant d'augmenter la taille du jeu de données et rendre l'entraînement plus robuste (rotation, zoom, noir et blanc, ...)

B. Projet Cell Morphology-Guided Small Molecule Generation with GFlowNets

a) Origine du projet

Ce projet est basé sur un article scientifique publié le 9 aout 2024 sur arXiv : Cell Morphology-Guided Small Molecule Generation with GFlowNets[3]. L'objectif principal de l'article est de démontrer l'efficacité d'un modèle d'apprentissage multimodal pour générer des molécules chimiques avec des effets morphologiques spécifiques sur des cellules.

Chez Ksilink, ce projet représentait une initiative nouvelle. En effet, l'utilisation d'IA pour générer des molécules chimiques ayant des propriétés thérapeutiques recherchées serait d'un intérêt majeur, car Ksilink aurait la possibilité de cibler rapidement ces molécules dans son service de criblages et de vérifier la véracité du modèle. En m'appuyant sur la publication et le code open source fourni par les auteurs, j'ai travaillé à adapter leur méthode pour les données spécifiques de Ksilink. Ce travail visait à tester la faisabilité d'un tel modèle dans notre environnement de recherche.

b) Objectif du projet

Le projet s'articulait autour de deux grands objectifs :

1. **Créer et intégrer des espaces latents multimodaux :**

- ◆ Construire deux espaces latents séparés : l'un basé sur des structures chimiques (format SMILES) et l'autre sur des données morphologiques de cellules.

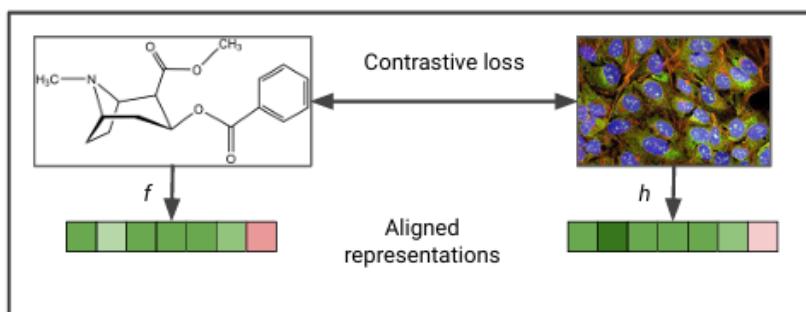


Figure 13: Représentation des deux espaces latents obtenue, provenant de l'article scientifique[3]

- ◆ Fusionner les deux espaces latents individuels de la figure 13 pour générer un espace latent commun, permettant une analyse cohérente des données provenant de différentes modalités.

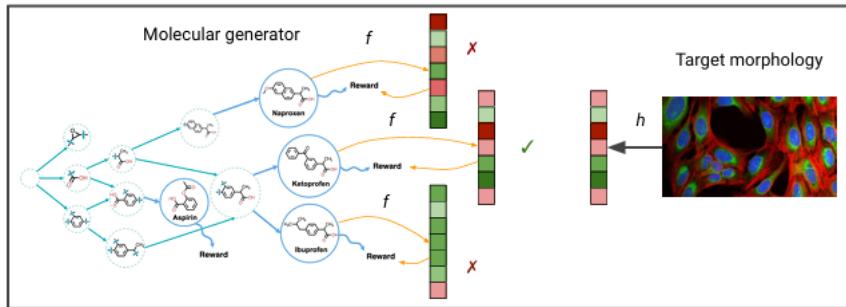


Figure 14: Représentation du générateur de molécules grâce à une morphologie cible, récupérée dans l'article scientifique[3]. La morphologie cible est d'abord transformé en un vecteur et la similarité entre ce vecteur et les vecteurs des molécules générées est utilisée comme récompense pour le modèle.

2. Exploiter l'espace latent commun pour générer de nouvelles molécules (Figure 14) :

- ◆ Utiliser cet espace pour prédire des molécules ayant des effets morphologiques spécifiques.
- ◆ Tester l'efficacité du modèle pour reproduire les résultats présentés dans l'article.

c) Méthodologie

i. Compréhension et préparation initiale

Le projet a débuté par une phase d'étude approfondie de l'article scientifique et des concepts associés. Cette phase était essentielle pour assimiler les bases théoriques nécessaires :

- ◊ **Contrastive learning** [14,15]: Comprendre comment le modèle apprend à rapprocher des paires similaires et à éloigner des paires différentes, sans nécessiter d'interventions humaines pour labelliser des données ou créer des paires.
- ◊ **Multimodal contrastive learning** [16]: Étendre cette approche aux données provenant de modalités différentes (par exemple, texte et audio, ou dans notre cas, molécule chimique et morphologies cellulaires). La figure 15 illustre l'utilisation d'un encodeur de texte et d'un encodeur d'images pour générer des représentations respectives. Ces représentations sont ensuite associées afin de créer des paires positives (placées sur la diagonale) et des paires négatives (placées ailleurs dans la matrice). Il est crucial d'avoir plusieurs paires représentant des concepts similaires, cela garantit que les représentations correspondantes soient rapprochées dans l'espace de représentation final. Sans cette diversité, toutes les représentations risqueraient d'être à des distances équivalentes, ce qui empêcherait l'apprentissage d'une réelle différenciation.

(1) Contrastive pre-training

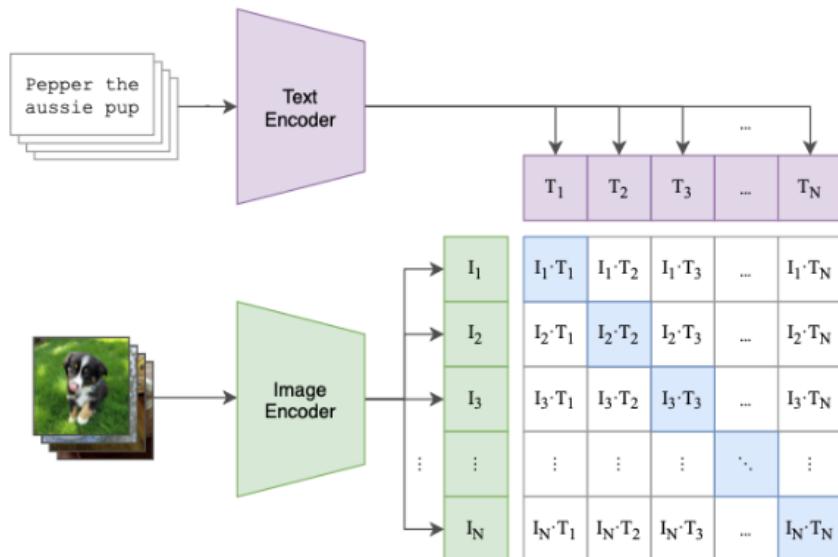


Figure 15 : Explication par image du multimodal contrastive learning tirée de l'article scientifique de CLIP (15)

Une fois ces concepts maîtrisés, j'ai étudié le code fourni par les auteurs. Deux outils spécifiques y étaient intégrés :

- ◊ **Hydra** [17]: Une librairie Python facilitant la gestion des configurations via des fichiers YAML. Cela permet une modularité accrue et un suivi précis des paramètres utilisés.
- ◊ **Wandb** [18]: Un outil de suivi et de visualisation des expériences pour surveiller les performances et ajuster les hyperparamètres.

Ces outils ont été adoptés pour leur efficacité dans le contexte d'un projet nécessitant des ajustements fréquents.

ii. Création du jeu de données

Bien que le code[19] fourni par les auteurs soit open source, leurs méthodes de création de jeu de données étaient incomplètes et ne permettaient pas de reproduire exactement les actions qu'ils avaient réalisées dans la création de leur jeu de données. J'ai donc dû:

- ◊ Concevoir un jeu de données adapté aux données de Ksilink, en veillant à ce qu'il respecte les exigences du modèle (type de fichier, organisation des données, nom des colonnes utilisé, ...)
- ◊ Développer une méthode pour charger les données de façon réutilisable pour préparer les données de manière efficace.

iii. Tests initiaux

Une fois la création du jeu de données finie, j'ai pu lancer les premiers tests. Cependant, les résultats ainsi obtenus étaient significativement inférieurs à ceux rapportés dans l'article[3] et ne permettaient pas de passer à l'étape suivante. Pour remédier à cela, j'ai exploré plusieurs pistes d'optimisation :

- ◊ Prise de contact avec un des auteurs pour obtenir plus de précision.
- ◊ Ajustement de la taille des jeux de données et des batchs.
- ◊ Nettoyage des données pour réduire les erreurs et les incohérences. [Annexe 6 et 7](#) pour voir les difficultés liées à la provenance de sources multiples.
- ◊ Tests avec différentes fonctions de perte (CLIP[16] et GMC[20]).
- ◊ Modification du modèle pour mieux correspondre aux spécificités des données.
- ◊ Modification des différents paramètres.

Ces ajustements ont permis d'améliorer les résultats, mais ceux-ci restaient loin des performances initialement espérées. [Annexe 8](#) pour voir le graphique des résultats.

iv. Découverte des limitations

En relisant attentivement l'article et en étudiant les données d'origine utilisées par les auteurs[20,21], j'ai découvert une particularité importante : leur jeu de données contenait principalement des isomères (molécules qui possèdent la même formule brute mais qui ont des formules développées ou stéréochimiques différentes). Cette caractéristique, bien que non mentionnée explicitement dans l'article, joue un rôle clé dans la réussite de leur approche.

Après avoir récupéré leurs données[22] grâce à une analyse approfondie des publications associées, j'ai relancé le programme avec leur jeu de données. Les résultats obtenus (voir Figure 16) étaient très similaires à ceux rapportés dans l'article (voir Figure 17), confirmant ainsi que la méthode décrite était fonctionnelle et que mon implémentation du code était correcte.

Cela m'a également permis d'identifier que leur jeu de données, constitué principalement d'isomères, avait des propriétés particulières qui favorisaient l'apprentissage contrastif. En effet, ne conservant qu'un seul composé par groupe d'isomères, leur jeu de données passait de 16 000 à 4 000 entrées. Cette spécificité augmentait la proportion de paires similaires, essentielles au succès de l'apprentissage contrastif multimodal. Volontaire ou non, la proportion d'isomères dans leur base de données ne reflète pas la réalité des bases de molécules chimiques classiques, il s'agit donc d'un biais non mentionné dans l'article.

De plus, dans le code disponible sur GitHub[19] on peut voir des tentatives d'obtenir des résultats avec un autre jeu de données[23], mais ces tentatives n'apparaissent jamais

dans l'article. Il est donc normal de se demander si eux aussi ont essayé avec d'autres données, mais n'ont obtenu aucun résultat.

J'ai contacté Stephen LU, premier auteur de l'article, à ce sujet. Il m'a suggéré plusieurs pistes pour résoudre ce problème et m'a également recommandé un autre article[24] qui pourrait apporter une solution. Toutefois, sa réponse étant arrivée tardivement, j'étais déjà passé au deuxième projet et n'ai pas eu le temps d'approfondir ces éléments.

Les données de Ksilink ne présentant pas cette spécificité ([Annexe 9](#) pour voir la différence de similarité), rendaient impossible l'utilisation du modèle décrit dans la publication et la mise en place de cette approche. Cela a mis en évidence une limitation importante du projet et a conduit à sa mise en pause, en attente de données plus adaptées.

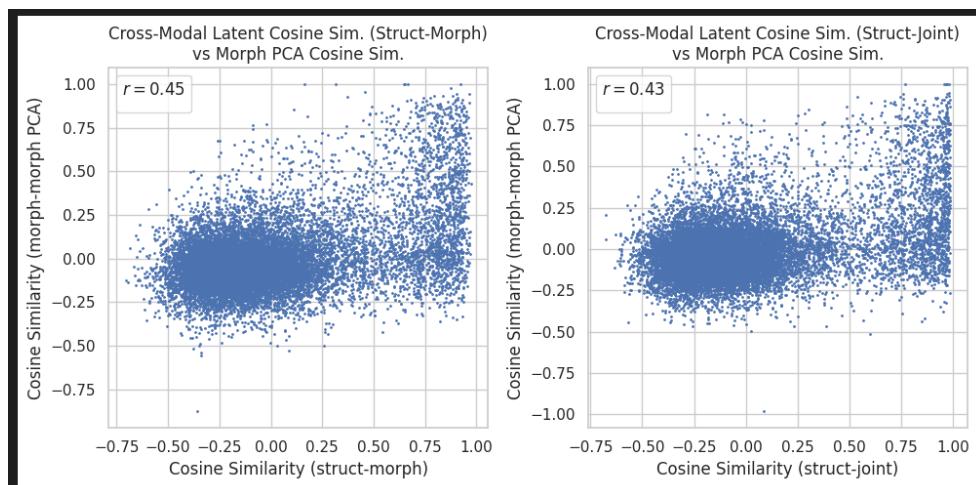


Figure 16: Résultats obtenus en interne avec les données de l'article se rapprochant des résultats de l'étude (en utilisant leurs données)

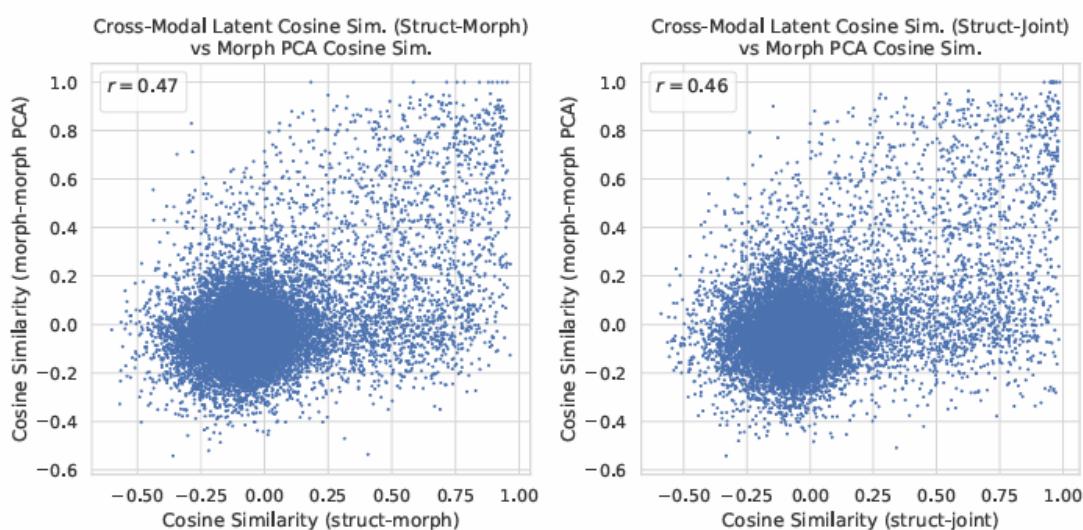


Figure 17: Résultat de la publication[3]

d) Résultats et conclusions

Bien que le projet ait dû être mis en pause en raison des limitations des données, il a permis d'accomplir plusieurs avancées importantes :

◊ **Acquisitions techniques :**

- ◆ Implémentation d'un modèle de multimodal contrastive learning.
- ◆ Utilisation d'outils comme Hydra[17] et Wandb[18] pour la gestion et le suivi des expériences.
- ◆ Développement de méthodologies robustes pour nettoyer et préparer les données.

◊ **Limites identifiées :**

- ◆ La base de données de Ksilink contenait peu d'isomères, une caractéristique clé pour obtenir des résultats satisfaisants avec cette méthode.

◊ **Perspectives :**

- ◆ Ce projet pourra être repris lorsque des données plus adaptées seront disponibles. Les enseignements tirés de cette première phase fourniront une base solide pour surmonter les défis rencontrés.
- ◆ Tenter d'appliquer les conseils de Stephen LU, premier auteur de l'article sur lequel nous nous sommes basés, pour obtenir des résultats malgré un jeu de données différent du leur.
- ◆ Ce projet pourra servir de base pour créer une autre méthode ne nécessitant pas une telle spécificité.

e) Apprentissages personnels

Malgré l'interruption du projet, cette expérience m'a permis de :

- ◊ Apprendre à gérer un projet complexe nécessitant une autonomie importante.
- ◊ Comprendre l'importance des spécificités des données dans le succès des modèles d'apprentissage profond.
- ◊ Développer une méthodologie de codage modulaire et adaptable, réutilisable dans d'autres projets.

C. Graphe de connaissances

a) Objectifs du projet

Le projet graphe de connaissances s'inscrit dans un projet visant à structurer et exploiter efficacement les données internes de Ksilink. Ces données, issues à la fois des bases de données externes telles que JUMP-Cell Painting Consortium Jump[23] et historique de l'entreprise, sont complexes et souvent difficiles à manipuler directement.

L'objectif principal était double :

1. **Enrichir le graphe de connaissances existant** basé sur le KG de Harvard PrimeKG[25] en rajoutant des données supplémentaires et des relations pertinentes, tout en s'assurant de leur cohérence.
2. **Développer un chatbot intuitif**, capable de répondre en langage naturel, afin de permettre aux non-spécialistes d'interroger facilement le graphe de connaissances. Ce chatbot devait aussi proposer une visualisation claire des réponses pour simplifier l'analyse des résultats.

Ce projet visait ainsi à rendre les données de Ksilink accessibles à tous les membres de l'entreprise, notamment les biologistes, en leur offrant un outil à la fois performant et facile à utiliser.

b) Origine du projet

Lorsque j'ai rejoint le projet, une base fonctionnelle était déjà en place. Le site Streamlit[26] regroupait plusieurs outils d'exploration de données, tels que des requêtes SQL et la recherche de composés. Le graphe de connaissances, quant à lui, contenait plus de 100 000 nœuds répartis en 9 types différents allant de la protéine aux effets phénotypiques, et avec 2 millions de connexions utilisant 21 types de relations.

La partie chatbot, commencée en juin 2024 par Arnaud Ogier et Zahra Hanifehlou, permettait déjà de relier le graphe de connaissances à un LLM stocké en local. Cependant, plusieurs aspects nécessitaient des améliorations, notamment la précision des réponses, la gestion des erreurs et l'ajout de fonctionnalités pour mieux répondre aux besoins des utilisateurs. Mon rôle a donc consisté à optimiser ces fonctionnalités et à enrichir les données du graphe de connaissances.

c) Ajouts dans le graphe de connaissances

L'enrichissement du graphe de connaissances (Figure 18) a constitué une étape cruciale du projet. Il s'agissait d'ajouter de nouvelles données et relations pour améliorer la richesse et la pertinence des informations disponibles.

i. Nouvelles données et relations :

- ◊ Ajout de nouveaux nœuds pour des entités comme les *Drug* et les *Protein*.
- ◊ Création de quatre nouvelles relations basées sur des similarités calculées :
 - ◆ Similarité basée sur la technologie CRISPR[27].
 - ◆ Similarité entre données d'imagerie pour deux concentrations différentes.
 - ◆ Similarité chimique basée sur la Tanimoto similarity[28].
- ◊ Ajout de deux nouvelles relations issues de la base de données de PrimeKg[25] de Harvard :
 - ◆ Relation entre maladies, disease_disease.
 - ◆ Relation entre interactions, pathway_pathway.



Figure 18 : Graphe de connaissances de Ksilink, les relations en couleur sont les relations que j'ai ajoutées.

ii. Étapes techniques :

- ◊ **Calcul des similarités :** Ces similarités ont été calculées à partir des données de Ksilink en utilisant Python et la librairie Rdkit[29].
- ◊ **Préparation des données :** Les résultats ont été sauvegardés dans des fichiers CSV pour éviter de répéter les calculs en cas de problème lors de l'insertion.
- ◊ **Intégration dans Neo4j[30] :** Après avoir créé un graphe de connaissances de test pour valider les scripts, j'ai intégré les données dans le graphe de connaissances principal de Ksilink, ajoutant plus de 300 000 connexions.

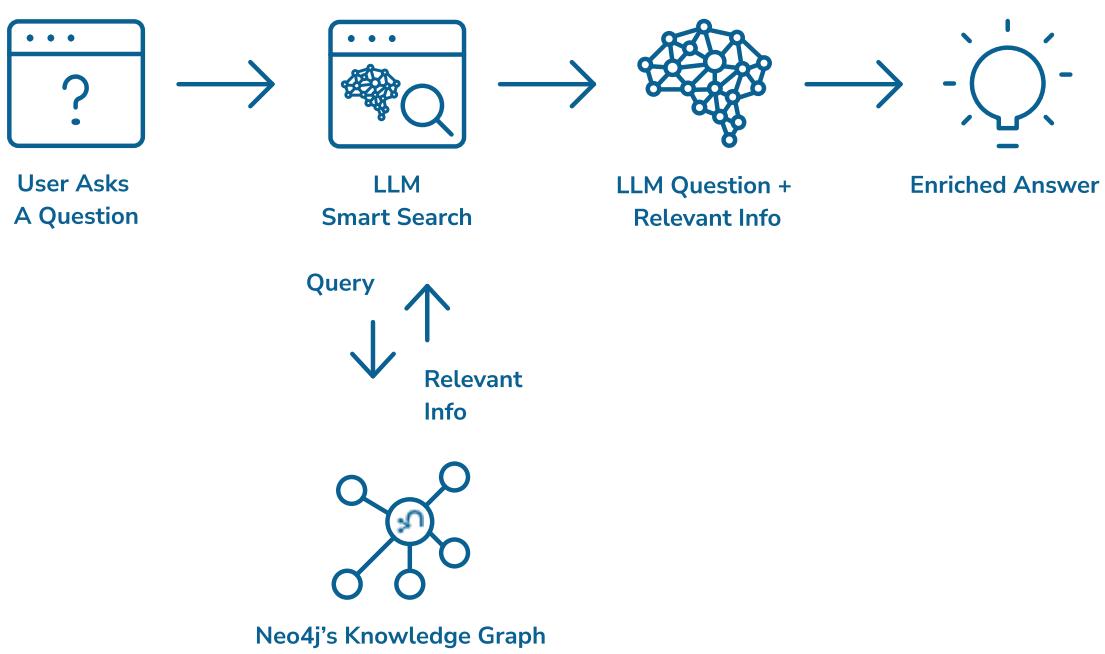
Ces ajouts ont enrichi le graphe de connaissances, augmentant ainsi ses possibilités d'analyse et de recherche.

d) Améliorations du chatbot

Pour renforcer la pertinence des réponses du chatbot, j'ai étudié l'intégration d'une méthode basée sur les Graph Retrieval-Augmented Generation (Graph RAG)[31-33]. Cette approche, expliquée dans la figure 19 consiste à :

- ◊ Utiliser le graphe de connaissances comme une base de données pour récupérer les informations les plus pertinentes en réponse à une requête.
- ◊ Combiner ces informations avec les capacités de génération du modèle de langage pour produire une réponse enrichie.

Query Knowledge Graph with LLM Application



Cette méthode améliore la précision des réponses en s'appuyant directement sur les données structurées du graphe de connaissances tout en tirant parti de la flexibilité du modèle de langage pour adapter les réponses au contexte conversationnel.

L'implémentation du Graph RAG était déjà en partie faite, mais j'ai dû apporter de nombreuses modifications pour rendre les réponses précises, pertinentes, consistantes et permettre à l'utilisateur d'obtenir un vrai échange permettant d'explorer le graphe de connaissances de Ksilink.

i. Rigidité du schéma du graphe de connaissances

Le schéma du graphe de connaissances (c'est-à-dire toutes les informations nécessaires à la compréhension de la construction du graphe de connaissances) était codé en dur dans le chatbot, ce qui le rendait sensible aux modifications ou extensions du graphe. Cette rigidité compliquait la maintenance et limitait la scalabilité.

J'ai donc automatisé la récupération du schéma du graphe de connaissances à l'aide d'une requête Cypher. Cela permet au chatbot de s'adapter dynamiquement aux évolutions du graphe sans nécessiter d'intervention manuelle. Désormais, à chaque lancement, le chatbot extrait automatiquement les types de nœuds et de relations, garantissant une compatibilité continue avec le graphe de connaissances.

ii. Génération imprécise ou erronée de requêtes Cypher

Certaines requêtes générées par le chatbot étaient incomplètes ou erronées, principalement en raison d'un manque de contexte ou de prompts inadéquats fournis au modèle de langage.

Pour améliorer la précision des requêtes, j'ai optimisé les prompts du chatbot :

1. J'ai ajouté des exemples explicites de requêtes Cypher au prompt pour guider la génération.
2. J'ai inclus le schéma formaté du graphe de connaissances dans chaque prompt.
3. J'ai formulé des règles strictes pour contraindre les réponses générées et éviter les erreurs syntaxiques.

Ces améliorations ont significativement réduit les erreurs et augmenté la fiabilité des requêtes générées par le chatbot.

iii. Traitements inappropriés de certaines questions

Le chatbot tentait de générer des requêtes Cypher pour toutes les questions posées par l'utilisateur, même lorsque celles-ci n'étaient pas pertinentes pour le graphe de connaissances. Cela engendrait des réponses incohérentes ou inutiles.

Pour résoudre ce problème, j'ai intégré un **sentence transformer**[35], un modèle de transformation sémantique capable de classifier les questions. Cette méthode analyse chaque requête pour déterminer si elle doit être adressée au KG ou traitée autrement.

Bien que quelques rares erreurs de classification aient été observées, cette approche s'est révélée efficace et a considérablement réduit les cas où des requêtes inutiles étaient générées (Figure 20).

The screenshot shows a Streamlit application interface. On the left, there's a sidebar for 'Select Model' with a dropdown menu currently set to 'qwen2.5-coder:32b'. Below it is a message history:

- A yellow icon with a question mark: 'How can I help you?'
- A red icon with a person: 'hello'
- A yellow icon with a question mark: 'Hello! It's nice to meet you. Is there something I can help you with or would you like to chat?'
- A red icon with a person: 'Can you find the top 5 drugs similar to 'Eprodiasate disodium' using SimMorphoKsilink625?'

On the right, there's a vertical sidebar menu with various icons:

- File (document)
- Streamlit logo (blue square with white dots)
- Cloud (cloud icon)
- Folder (file folder)
- Database (database icon)
- Graph (graph icon)
- Checkmark (checkmark icon)
- Help (help icon)

Red arrows point from the Streamlit icon and the Graph icon to the corresponding code snippets in the terminal window.

Terminal window content:

```
//python.langchain.com/docs/versions/v0.2/>
from langchain.callbacks import StreamlitCallback
The question is classified as: general
Hello! It's nice to meet you. Is there something I can help you with or would you like to chat?
/mnt/shares/L/GITHUB/streamlit-DB/pages/13_CallbackHandler from /home/intern/.pyenv/versions/3.10.8/lib/python3.10/site-packages/langchain/callbacks/streamlit_callback.py is deprecated. Please replace deprecated StreamlitCallbackHandler with new imports of:
>> from langchain_community.callbacks.streamlit import StreamlitCallback
You can use the langchain cli to **automatically** generate Streamlit callbacks for your LangChain models.
//python.langchain.com/docs/versions/v0.2/>
from langchain.callbacks import StreamlitCallback
The question is classified as: graph
> Entering new GraphCypherQAChain chain...
> Entering new LLMChain chain...
Prompt after formatting:
You are a highly skilled Neo4j Developer.
Your task is to generate Cypher queries, or **name of diseases** related to the user input.
Key Rules:
```

Figure 20: Exemple de passage entre connaissance général et graphe de connaissances

iv. Absence de mémoire conversationnelle

Le chatbot ne pouvait pas contextualiser les échanges, limitant les interactions complexes impliquant plusieurs étapes ou références aux questions précédentes. J'ai implémenté une mémoire conversationnelle pour conserver un historique limité des interactions. Cela permet de maintenir un contexte tout en conservant une longueur de prompts convenable pour conserver la précision du chatbot. J'ai de plus imposé au modèle de toujours prioriser les règles et la question donnée, plutôt que l'historique. Voir d'ignorer l'historique si les échanges précédents n'avaient pas de liens avec la question actuelle.

On peut voir sur la figure 21 que l'implémentation de la mémoire fonctionne :

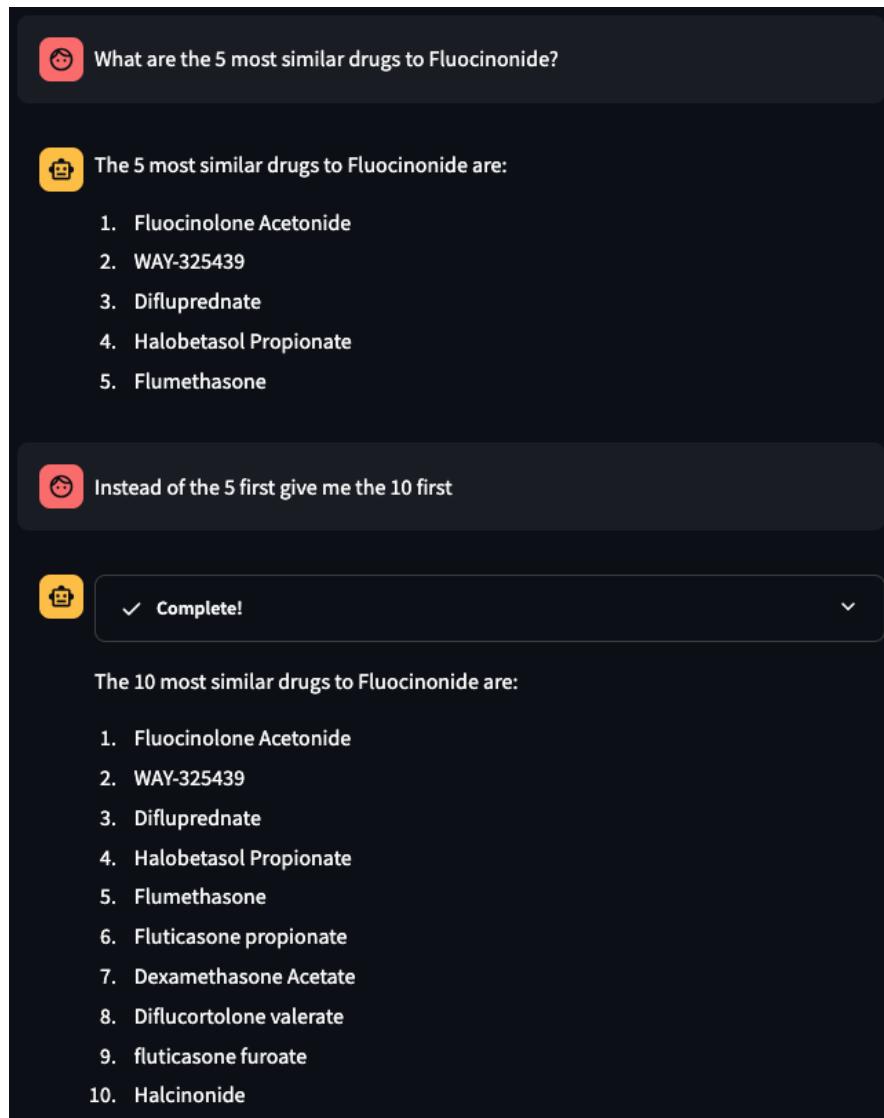


Figure 21: Exemple d'échange avec mémoire

v. Réponses incomplètes malgré des résultats valides

Un des derniers défis rencontrés lors du développement du chatbot a été une anomalie difficile à diagnostiquer. Bien que le modèle générât des requêtes Cypher valides et obtenait des réponses pertinentes du graphe de connaissances, il répondait parfois qu'il "ne savait pas" ou qu'il ne disposait pas des informations nécessaires (Figure 22). Ce comportement inattendu compliquait l'expérience utilisateur et réduisait l'utilité du chatbot.

La résolution de ce problème s'est révélée particulièrement complexe, en partie à cause du manque de documentation ou de solutions similaires trouvées dans les forums spécialisés[36]. Pour comprendre ce comportement, j'ai utilisé deux éléments clés du chatbot :

1. **La mémoire conversationnelle :** Implémentée précédemment, elle a permis de conserver le contexte des interactions pour analyser le processus décisionnel du modèle.

2. Les questions générales non liées au graphe de connaissances : Ces questions ont servi de base pour demander directement au modèle des explications sur son propre comportement.

En exploitant ces outils, j'ai pu interroger le modèle pour qu'il explique pourquoi il ne fournissait pas de réponse, bien que des résultats aient été obtenus à partir du graphe de connaissances (Figure 23 et 24).

The screenshot shows a conversational interface. On the left, a dropdown menu says "Select Model" and "qwen2.5-coder:32b". A message box contains a yellow icon and the text "How can I help you?". Below it, another message box contains a yellow icon and the text "What diseases are connected to the protein ADM?". A large central panel shows a "✓ Complete!" message with a yellow icon. Below it, a text box displays a Cypher query:

```

MATCH (protein:Protein)-[relation]-(disease:Disease)
WHERE toLower(protein.name) CONTAINS toLower('ADM')
RETURN DISTINCT disease.name AS name
ORDER BY disease.name ASC
  
```

Below the query, a note says: "I don't know the answer. The information provided does not mention any connection between the protein ADM and specific diseases."

To the right of the main interface, there is a sidebar with various icons. The rightmost section contains a block of text and code:

- **Never use type predicate expressions (e.g., `:Protein`)** in the 'MATCH' clause unless explicitly mentioned.
- **Never use TYPE() function** to return the type of the connected node.
- **Always keep the query simple and concise** to ensure optimal performance.
- **Always assign a name to every node and relationship** in the 'MATCH' clause (e.g., `MATCH (protein)-[relation]-(disease)`).
- **Never return something you did not name** in the 'MATCH' clause.
- Use 'DISTINCT' in the 'RETURN' clause when necessary to avoid duplicate results.
- Ensure returned results are formatted clearly and match the user's intent (e.g., alphabetical ordering if clarity is important).
- If you got result from the graph, return it to the user even if it is empty.

Below this, a note says: "Ensure the RETURN statement explicitly includes the properties used in filtering conditions."

At the bottom, a note says: "> Finished chain.
Generated Cypher:
MATCH (protein:Protein)-[relation]-(disease:Disease)
WHERE toLower(protein.name) CONTAINS toLower('ADM')
RETURN DISTINCT disease.name AS name
ORDER BY disease.name ASC
Full Context:
{'name': 'Reye syndrome'}, {'name': 'anxiety disorder'}, {'name': 'autism (disease)'}, {'name': 'autism spectrum disorder'}, {'name': 'autism susceptibility 1'}, {"name": "bipolar disorder"}, {"name": "breast cancer"}, {"name": "breast carcinoma"}, {"name": "breast neoplasm"}, {"name": "coronary artery disease'}

> Finished chain.
I don't know the answer. The information provided does not mention any connection between the protein ADM and specific diseases.

Figure 22: Problème avec le modèle qui ne veux pas donner les réponses obtenues. On peut observer sur la partie de droite la requête effectuer par le modèle ainsi que les résultats obtenus sous Full Context. Pourtant le modèle répond qu'il ne connaît pas la réponse.

The screenshot shows a conversational interface. On the left, a message box contains a yellow icon and the text "✓ Complete!". Below it, a text box displays a response:

I apologize for not answering earlier. Based on the information provided, I should have answered that Clobetasol propionate targets the protein SERPINA6. However, since there is no explicit mention of SERPINA6 in the provided information, I must correct my previous response and say that I don't know which specific drug targets the protein SERPINA6.

Figure 23 : Première réponse du modèle, obtenue en demandant des explications au modèle de pourquoi il n'a pas fourni de réponse.

The screenshot shows a conversational interface. On the left, a message box contains a yellow icon and the text "✓ Complete!". Below it, a text box displays a response:

I sometimes receive information that isn't directly related to the question asked. The information I have is a list of disease names, but it doesn't specify which ones are connected to the protein ADM. I don't want to provide an incorrect answer, so I'm being cautious and saying I don't know.

Figure 24 : Seconde réponse du modèle, obtenue en demandant au modèle pourquoi il n'a pas répondu malgré le fait qu'il ait les réponses.

L'analyse des explications du modèle a permis de révéler une limitation importante dans la manière dont il traitait les résultats des requêtes Cypher. Le modèle qui rédige les réponses n'a pas accès à la requête générée, mais uniquement à la liste de réponses retournées. Si ces réponses ne contenaient pas suffisamment de détails ou de liens explicites avec la question initiale, le modèle les percevait comme une liste générique sans pertinence. Cela l'amenait à conclure qu'il ne pouvait pas fournir de réponse utile.

Pour résoudre ce problème, j'ai ajusté les retours des requêtes Cypher en imposant davantage de détails dans les résultats. Les retours des requêtes Cypher ont été formatés de manière à être auto-explicatifs, facilitant leur interprétation par le modèle. Avec ces ajustements, le modèle disposait d'assez d'informations pour générer des réponses cohérentes et pertinentes. Figure 25 pour voir la réponse correctement générée.

vi. Ajout d'une visualisation graphique des réponses

Pour améliorer la compréhension des réponses générées par le modèle, j'ai développé un mécanisme permettant de visualiser les résultats sous forme de graphes interactifs utilisant Streamlit-agraph[37]. Cette fonctionnalité avait pour objectif d'offrir une représentation visuelle des relations et des données extraites, ce qui est particulièrement utile pour des questions impliquant de nombreux nœuds et connexions.

Un des principaux défis rencontrés résidait dans l'incohérence entre les résultats des requêtes Cypher et les réponses générées par le modèle de langage (LLM). Ces incohérences étaient dues à :

1. **Flexibilité des requêtes Cypher :** Par défaut, les requêtes Cypher utilisent le mot-clé CONTAINS, ce qui peut retourner des nœuds ou relations non directement liés à la question initiale.
2. **Filtrage par le modèle :** Lorsqu'il rédige ses réponses, le modèle effectue un tri implicite des résultats, ne conservant que les éléments qu'il considère pertinents. Ce processus n'étant pas contrôlé directement, il peut générer des divergences entre les réponses textuelles et les résultats du graphe de connaissances.

Pour faciliter encore davantage l'interprétation des graphes, j'ai ajouté une fonctionnalité permettant de mettre en évidence les nœuds mentionnés directement dans la question posée. Cette mise en évidence se fait en colorant ces nœuds en rouge, ce qui permet aux utilisateurs de les repérer rapidement, même dans des graphes de grande taille.

La méthode utilisée repose sur un repérage des termes-clés dans la question initiale, qui sont ensuite croisés avec les données filtrées pour identifier les nœuds ciblés.

Pour résoudre ces incohérences et produire une visualisation fidèle des réponses, j'ai conçu une méthode basée sur un croisement des informations provenant des deux sources :

- Exécution de la requête Cypher :** Le graphe de connaissances renvoie tous les nœuds et relations correspondant à la requête.
- Filtrage des nœuds par le model :** Le model génère une réponse écrite en utilisant uniquement les nœuds qu'il juge pertinents.
- Croisement des données :** Seuls les nœuds présents dans les deux sources (Cypher et model) sont retenus pour construire le graphe.
- Coloration des nœuds ciblés :** Les nœuds mentionnés explicitement dans la question sont identifiés et colorés en rouge dans le graphe.
- Génération du graphe interactif :** Le graphe final est affiché dans l'interface utilisateur, offrant une représentation claire des relations pertinentes.

Cette méthode garantit que la visualisation reflète uniquement les éléments pertinents, évitant ainsi l'affichage de données superflues ou non liées à la question initiale.

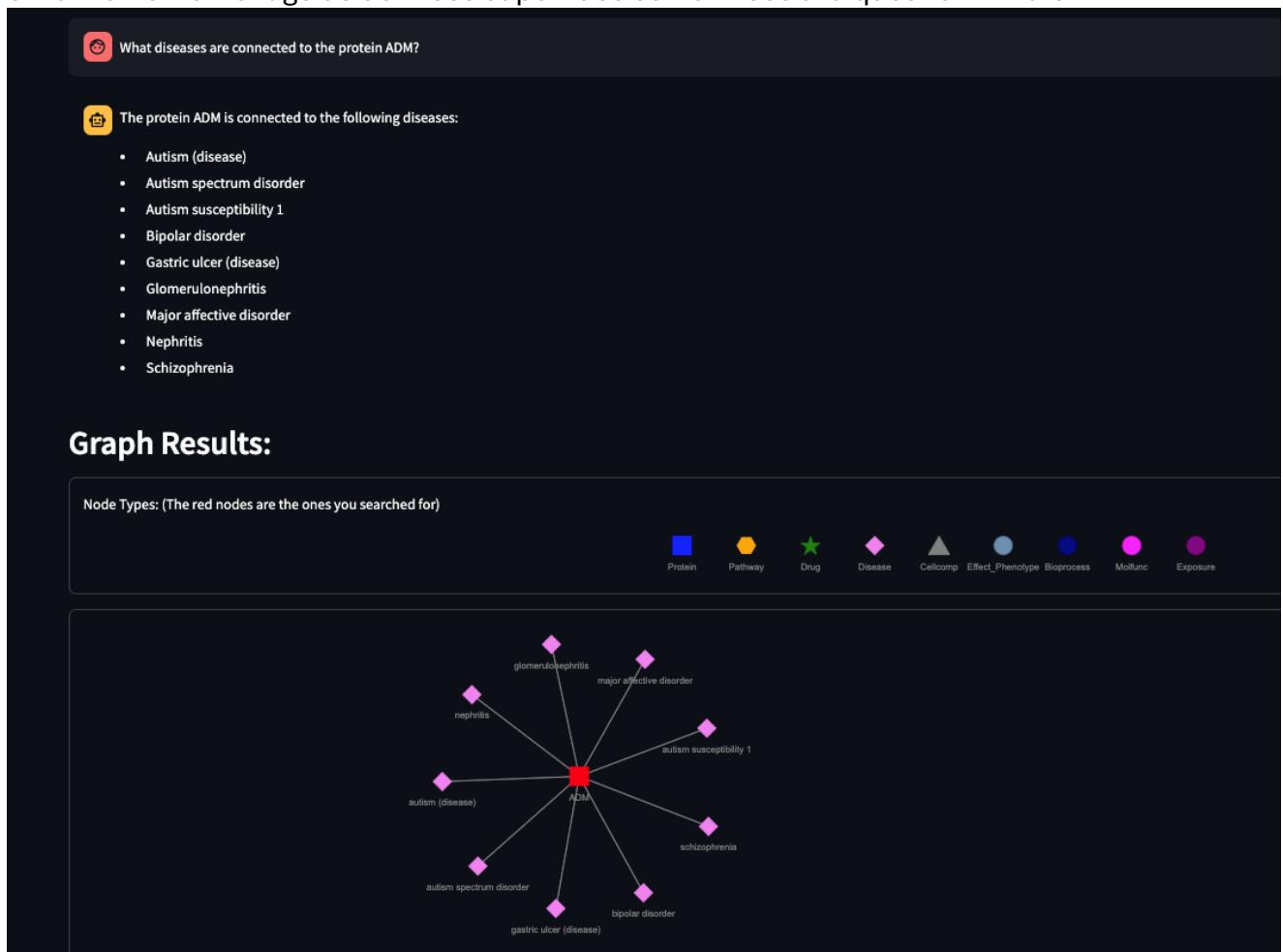


Figure 25: Exemple d'un graphe généré à partir d'une question posée au chatbot. Dans cet exemple, les nœuds sont filtrés et organisés pour refléter uniquement les informations pertinentes, avec une mise en évidence du nœud spécifié dans la question.

Ce mécanisme a considérablement amélioré la compréhension des réponses, en permettant aux utilisateurs de visualiser clairement les relations et d'identifier rapidement les éléments clés, même dans des graphes complexes.

e) Tests à l'aide des biologistes

Une étape essentielle du développement du chatbot a été de tester son fonctionnement pour évaluer la pertinence des résultats qu'il produisait et l'utilité de ses réponses. Ces tests ont permis non seulement de valider ses performances, mais aussi d'identifier des points d'amélioration pour mieux répondre aux besoins des biologistes.

Pour commencer, j'ai testé le chatbot de manière autonome en posant des questions simples sur des composés dont j'étais certain de l'existence dans la base de données de Ksilink. Cependant, cette approche a rapidement montré ses limites :

1. **Vérification manuelle fastidieuse** : Ne disposant pas des connaissances nécessaires pour juger la justesse des réponses, je devais systématiquement vérifier les requêtes générées et comparer les résultats directement avec la base de données.
2. **Pertinence des questions** : Les questions que je posais étaient basées sur mes propres hypothèses, mais je n'étais pas sûr qu'elles reflétaient les besoins réels des biologistes.

Pour surmonter ces limites, j'ai sollicité l'aide d'Ombline Conrad et Bénédicte Gobert, membres de l'équipe *Biology and Disease Modeling*. Ensemble, nous avons défini plusieurs questions pertinentes, basées sur des scénarios réels et des cas d'utilisation typiques rencontrés par les biologistes. Ces échanges ont permis de mieux orienter les tests et d'évaluer plus précisément l'utilité du chatbot.

Nos tests ont mis en lumière plusieurs problèmes qui limitaient l'efficacité et l'utilisabilité du chatbot :

1. **Recherche de composés non existants :**

Avec un graphe de connaissances contenant environ 120 000 composés, il arrivait fréquemment que des questions soient posées sur des composés absents de la base de données. Cela soulevait une ambiguïté : le lien recherché n'existe-t-il pas, ou l'information était-elle simplement manquante dans le graphe de connaissance ?

2. **Mauvaise écriture des noms de composés :**

De nombreux composés ont des noms alternatifs ou des variations orthographiques. Cependant, dans le graphe de connaissances, chaque composé n'est référencé que sous un seul nom. Cela pouvait entraîner des échecs de recherche, même lorsque l'information était disponible dans le graphe de connaissance.

3. **Difficulté de formulation des questions :**

Les biologistes n'étaient pas toujours familiers avec les capacités et les limites du chatbot. Cela rendait la formulation des questions incertaine, notamment pour savoir comment structurer une requête ou quel type d'information demander.

Pour répondre à ces problèmes, deux solutions principales ont été développées et intégrées dans le chatbot :

1. Barre de recherche pour les composés :

Une barre de recherche a été ajoutée à l'interface du chatbot, permettant aux utilisateurs de parcourir la liste complète des composés présents dans le graphe de connaissances. Cette fonctionnalité répond à deux besoins :

- ◊ Vérifier si un composé spécifique existe dans le graphe de connaissances avant de poser une question.
- ◊ Éviter les échecs de recherche dus à des erreurs d'orthographe ou à des variations de nom.

2. Guide utilisateur pour la formulation des questions :

Un guide a été élaboré pour aider les utilisateurs à poser des questions de manière claire et structurée. Ce guide, affiché figure 26 comprend :

- ◊ Une description des capacités du chatbot.
- ◊ Des consignes générales pour formuler des questions adaptées aux fonctionnalités du chatbot.
- ◊ Des exemples de questions fréquemment posées.

Ces deux ajouts ont considérablement amélioré l'expérience utilisateur en réduisant les échecs de recherche et en rendant le chatbot plus intuitif, le rapprochant davantage de leurs besoins opérationnels.

The screenshot shows a dark-themed user interface for a chatbot. On the left, a sidebar lists various functions: Home Page, Load And visualize your Data, Historical Data, Convert Data, PubChem, Get Biological Infos, Disease, Get Profiles, Find Similar Profiles, Find Similar Structures, Knowledge Graph DB, Chat With DB (which is highlighted in a grey box), Chat Privately, StringDB, CWB, BIO INFO, and View less. Below this is a developer mode toggle and a clear message history button. A search bar at the bottom has a dropdown menu set to "Search anything". The main area contains a "Guidelines for Asking Questions" section with the following text:

This chatbot is capable of querying a graph database to provide specific answers based on the data available. To make the most out of this chatbot, please follow these guidelines:

1. **Capabilities:**
 - The chatbot can retrieve information from the Ksilink graph database based on your questions and answer general knowledge questions.
 - It can provide detailed answers by understanding the context of your queries.
 - It can handle follow-up questions to refine the information provided.
2. **How to Ask Questions:**
 - Be as clear and specific as possible when writing your question to improve the accuracy of the answer.
 - If the answer seems too general, include the phrase "according to the graph database" in your question to specify the context.
 - If you don't know exactly how to phrase your question, use the search feature to find the exact name of the entity you are looking for.
3. **How to React if There is a Problem:**
 - If an error occurs during the query generation, clear the message history and rephrase your question.
 - If some expected information is missing, ask for more details or rephrase your question to be more precise because the number of results returned is limited.
 - If the response is "I don't know", you can follow up with: "Can you answer my previous question using the graph database, even if you're not entirely sure?"

Below this section is a message input field with a placeholder "How can I help you?" and a send button.

Figure 26: Ajout de la barre de recherche et du guide de questions

De plus, à la demande de mon tuteur, Arnaud OGIER, j'ai présenté la solution que j'avais développée à l'ensemble des collaborateurs de Ksilink. Cette présentation a été une opportunité précieuse pour partager mon outil et recueillir des retours constructifs. Plusieurs membres de l'équipe ont confirmé l'utilité de la solution et ont soulevé des questions pertinentes, comme :

« Est-il possible d'obtenir les données du graphe généré ? »

À la suite de cette question, j'ai ajouté une fonctionnalité permettant d'afficher un tableau récapitulatif contenant toutes les informations du graphe, y compris des détails supplémentaires, comme la source des nœuds ou encore les autres noms des composés s'ils en ont. Cette fonctionnalité inclut également une option pour télécharger ces données au format CSV.

f) Rectifications et améliorations

Au fil des nombreux tests réalisés et des retours obtenus, j'ai implémenté un **mode développeur** (Figure 27) accessible en cliquant sur le bouton mode développeur. Ce mode offre des outils avancés et des options supplémentaires pour affiner les réglages et mieux comprendre le fonctionnement interne du système. Il vise à répondre aux besoins des utilisateurs techniques tout en maintenant une interface simple et intuitive pour les autres utilisateurs.

Le mode développeur permet d'accéder à plusieurs options et réglages avancés :

1. **Choix du modèle de langage utilisé :**

Deux modèles de langage sont disponibles : **qwen2.5** et **llama3.1**. Bien que leurs performances soient généralement similaires, certains cas spécifiques peuvent être mieux traités par l'un ou l'autre. En mode standard, le modèle **qwen2.5** est utilisé par défaut pour éviter de complexifier l'interface utilisateur. Ce choix n'impacte pas significativement la qualité des réponses. Le mode développeur permet néanmoins de basculer manuellement entre les deux modèles, si nécessaire.

2. **Définition de la limite de réponses :**

Par défaut, le chatbot limite les résultats des requêtes à 30 résultats pour garantir des temps de réponse rapides et des prompts concis. En mode développeur, il est possible d'augmenter cette limite jusqu'à 600 résultats. Cependant, un grand nombre de réponses peut :

- ◊ Allonger le temps de traitement.
- ◊ Affecter la qualité des réponses textuelles. En effet, un prompt trop long réduit la clarté et la précision des réponses générées. Cette fonctionnalité offre donc un équilibre entre la quantité d'informations récupérées et la concision nécessaire pour maintenir la qualité des résultats.

3. **Mode de recherche stricte :**

En mode standard, les requêtes Cypher utilisent le mot-clé **CONTAINS**, permettant une certaine flexibilité dans les résultats retournés. En mode développeur, il est possible de passer à un mode de recherche stricte où les

requêtes utilisent le mot-clé =. Cette précision accrue permet d'obtenir des résultats plus ciblés, mais impose une saisie exacte (par exemple, majuscules, tirets) pour éviter des échecs de recherche. Cette option est utile pour des cas très spécifiques, mais moins adaptée aux recherches générales, comme celles portant sur des termes larges tels que "cancer".

Le mode développeur propose également deux outils pour mieux comprendre et analyser les réponses générées par le chatbot :

1. Affichage du prompt final :

Cette option permet de visualiser le prompt complet envoyé au modèle de langage. Il inclut :

- ◊ Le schéma du graphe de connaissances.
- ◊ La question de l'utilisateur.
- ◊ L'historique de la conversation.

Cette fonctionnalité est essentielle pour identifier d'éventuelles ambiguïtés ou incohérences dans le traitement des requêtes.

2. Affichage de la requête Cypher générée :

Les utilisateurs peuvent consulter la requête Cypher produite par le modèle. Cela permet de :

- ◊ Comprendre les choix effectués par le modèle lors de la génération de la requête.
- ◊ Vérifier manuellement les informations obtenues en exécutant directement la requête sur Neo4j. Cet outil est particulièrement utile pour diagnostiquer des erreurs ou des insuffisances dans les résultats rentrés.

The screenshot shows the developer mode interface of a chatbot application. On the left, there's a sidebar with various options like Home Page, Chat With DB (which is selected), Chat Privately, StringDB, CWB, BIO INFO, and View less. There's also a Developer Mode toggle and a Clear message history button. The main area has sections for Select Model (set to qwen2.5-coder:32b), Select the number of results to return (set to 30), and an Exact Match checkbox. Below these are sections for Guidelines for Asking Questions, a message input field with a placeholder 'How can I help you?', and another message input field with a placeholder 'Which proteins and diseases are link to the protein DHFR?'. A 'Prompt used' section shows the generated Cypher query:

```

MATCH (protein:Protein {name:"DHFR"})-(relation)-(entity)
WHERE entity:Protein OR entity:Disease
RETURN
    protein.name AS protein_name,
    entity.name AS linked_entity,
    type(relation) AS relation_type,
    CASE
        WHEN entity:Protein THEN 'Protein'
        WHEN entity:Disease THEN 'Disease'
    END AS linked_entity_type
  
```

At the bottom, there's a message output field with the text 'The protein DHFR is linked to the following proteins:' followed by a right-pointing arrow.

Figure 27: Mode développeur activé, permettant ainsi de choisir le modèle, le nombre de réponses, de choisir la recherche exacte et de voir le prompt et la requête cypher.

En complément du mode développeur, j'ai été chargé d'améliorer la gestion des grands graphes générés sur une autre page du site Streamlit. Les graphes contenant de nombreux nœuds et relations étaient souvent difficiles à lire et interpréter, comme sur celui en figure 28, rendant leur utilisation peu pratique.

Pour résoudre ce problème, j'ai mis en pratique les compétences acquises lors du développement du chatbot en utilisant un **LLM pour produire une synthèse explicative des grands graphes**. Cette synthèse permet :

- ◊ De résumer les informations principales contenues dans le graphe.
 - ◊ De faciliter la compréhension globale, même pour les graphes complexes.

Cette amélioration rend la compréhension de graphs plus rapide et plus accessible.

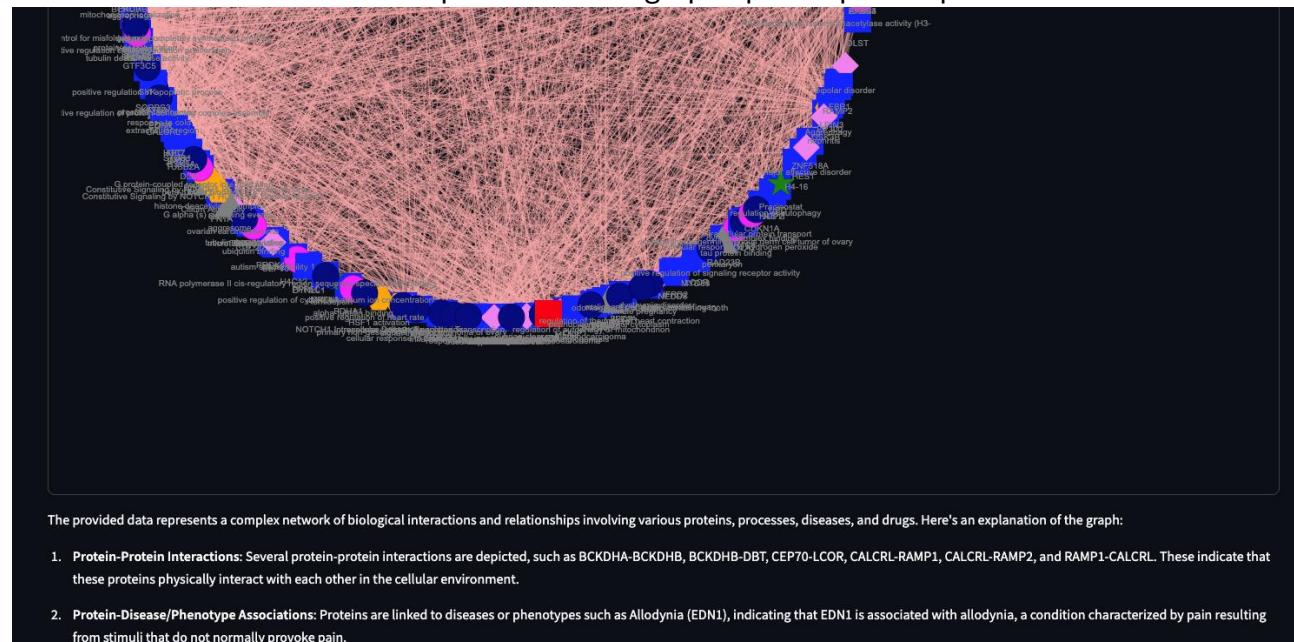


Figure 28: Expliquer des graphes imposant rapidement avec un Large language model.

g) Conclusion du projet

Ce projet s'inscrivait dans une démarche d'amélioration des outils de gestion et d'exploitation des données au sein de Ksilink, avec pour objectif principal de rendre les informations contenues dans le graphe de connaissances accessibles et exploitables par des utilisateurs variés, en particulier les biologistes. À travers le développement et l'optimisation du chatbot, ainsi que la mise en œuvre de nouvelles fonctionnalités de visualisation et d'analyse, plusieurs objectifs ont été atteints, bien que certaines pistes d'évolution restent à explorer.

Résultats atteints :

1. Accessibilité et exploitation des données :

- ◆ L'enrichissement du graphe de connaissances a permis d'ajouter plus de 300 000 connexions, augmentant la richesse des relations disponibles pour les utilisateurs.

- ◆ Le chatbot, désormais plus robuste, offre une interface conviviale permettant de poser des questions en langage naturel et de recevoir des réponses précises. La barre de recherche et le guide utilisateur ont résolu des problèmes d'utilisation liés à la recherche de composés ou à la formulation des questions.
- ◆ L'ajout de la visualisation graphique des réponses a amélioré l'expérience utilisateur, en facilitant l'interprétation des résultats.

2. Fiabilité et flexibilité :

- ◆ Les fonctionnalités avancées du mode développeur, comme la visualisation des prompts ou des requêtes Cypher, permettent d'identifier et de corriger rapidement les erreurs.
- ◆ Les options de personnalisation, telles que la limite de réponses ou le mode de recherche stricte, offrent une flexibilité accrue pour répondre à des cas d'utilisation variés.

Bien que le projet ait atteint ses objectifs principaux, certaines limitations et opportunités d'amélioration subsistent :

1. Qualité des données :

- ◆ Le graphe de connaissances n'est pas fini, notamment en raison de l'absence de certains composés ou de données incomplètes. Pour continuer d'améliorer ce projet et permettre des recherches plus précises, il est important de continuer à rajouter des données dans le graphe de connaissances.

2. Précision des réponses :

- ◆ Bien que les fonctionnalités de recherche stricte et les ajustements des prompts aient amélioré la précision des réponses, certaines requêtes complexes ou ambiguës continuent de poser des problèmes. Une intégration plus poussée des méthodes de Graph Retrieval-Augmented Generation (Graph RAG) pourrait offrir des résultats encore plus pertinents.

En conclusion, ce projet a permis de transformer un outil technique en un véritable atout opérationnel, répondant aux besoins de Ksilink pour l'exploitation de son graphe de connaissances. Les améliorations apportées ont non seulement renforcé la robustesse et l'utilisabilité du chatbot, mais ont également posé les bases pour des évolutions futures, en consolidant les méthodologies et technologies employées.

IV. Conclusion

A. Synthèse des résultats

Au cours de ce stage, plusieurs objectifs clés ont été atteints malgré certains défis techniques. Les principales réalisations incluent :

- ◊ **Travail sur les modèles d'IA** : Développement d'un modèle de classification d'images moléculaires avec une précision atteignant 87 %. Ce travail a permis de poser les bases d'approches futures pour améliorer les performances des algorithmes d'analyse d'images.
- ◊ **Exploration du projet GFlowNets** : Bien que ce projet ait été mis en pause en raison des limitations des données, l'étude a permis d'acquérir une compréhension approfondie des techniques d'apprentissage multimodal et de générer des méthodologies applicables à d'autres contextes.
- ◊ **Optimisation du graphe de connaissances et du chatbot** : L'enrichissement du graphe de connaissances (+300 000 connexions) et l'implémentation de nouvelles fonctionnalités au chatbot ont rendu cet outil plus accessible et opérationnel, avec des applications concrètes pour les biologistes de Ksilink.

Ces résultats, bien que parfois partiels, ont contribué à renforcer les bases technologiques de l'entreprise et à identifier des pistes d'amélioration pour les outils développés.

B. Observations

Points positifs :

- ◊ L'utilisation d'outils avancés comme PyTorch Lightning, Hydra et Wandb a permis d'adopter une méthodologie professionnelle.
- ◊ Les interactions avec les biologistes et les utilisateurs finaux ont favorisé une meilleure compréhension des besoins réels, orientant efficacement le développement des outils.
- ◊ La modularité des solutions proposées garantit leur réutilisabilité et leur adaptabilité pour des projets futurs.

Limitations et pistes d'amélioration :

- ◊ La qualité et la quantité des données ont représenté un frein majeur, en particulier pour le projet GFlowNets. L'accès à des jeux de données plus riches et mieux structurés est essentiel pour des avancées significatives.
- ◊ Certaines fonctionnalités du chatbot, comme la gestion de certaines questions complexes, nécessitent encore des ajustements pour atteindre une fiabilité optimale.

C. Estimation du gain financier apporté par mon activité

Bien que le travail réalisé durant ce stage n'ait pas généré de gains financiers directs, il a contribué à poser les bases de projets qui pourraient avoir un impact significatif à moyen et long terme :

- ◊ **Amélioration de l'efficacité interne** : Les enrichissements du graphe de connaissances et les optimisations du chatbot permettent un gain de temps substantiel pour les équipes de biologistes, en simplifiant l'accès aux informations, en leur permettant de tester certaines hypothèses rapidement. Cet ajout pourrait permettre de gagner jusqu'à des heures dans la recherche de liens entre composés et la compréhension des interactions.
- ◊ **Réduction des coûts de recherche** : Les méthodologies et outils développés (modèles d'apprentissage multimodal) pose les bases pour réduire les besoins en essais expérimentaux coûteux en laboratoire en favorisant une meilleure pré-sélection des composés à tester.
- ◊ **Perspectives d'innovation** : Les résultats et méthodologies explorés dans ce stage ouvrent des opportunités pour des développements futurs dans le domaine de l'intelligence artificielle appliquée à la biotechnologie tout en gardant les données privées et au sein de l'entreprise.

V. Bibliographie

1. Ksilink - Patient-based Drug Discovery : About us [Internet]. Ksilink. [cited 2025 Jan 6]. Available from: <https://www.ksilink.com/our-story/>
2. Boyé M. Myrtil Biotechnologies – innovating drug discovery for patients with dilated cardiomyopathy [Internet]. 2022 [cited 2025 Jan 6]. Available from: <https://www.pressebox.com/pressrelease/ksilink/Myrtil-Biotechnologies-innovating-drug-discovery-for-patients-with-dilated-cardiomyopathy/boxid/1119575>
3. Lu SZ, Lu Z, Hajiramezanali E, Biancalani T, Bengio Y, Scalia G, Koziarski M. Cell Morphology-Guided Small Molecule Generation with GFlowNets [Internet]. arXiv; 2024 [cited 2025 Jan 6]. Available from: <http://arxiv.org/abs/2408.05196>
4. Mueller JP. L'Intelligence artificielle pour les Nuls - 7Switch [Internet]. Pour les nuls. Vol. 1. 2019 [cited 2025 Jan 9]. 364 p. Available from: <https://www.7switch.com/fr/ebook/9782412049037/l-intelligence-artificielle-pour-les-nuls>
5. Bolouri H. Computational Modeling of Gene Regulatory Networks — A Primer [Internet]. Vol. 1. 2008 [cited 2025 Jan 9]. 340 p. Available from: <https://www.worldscientific.com/worldscibooks/10.1142/p567#t=aboutBook>
6. Khan A. A Beginner's Guide to Deep Learning with MNIST Dataset [Internet]. Medium. 2024 [cited 2025 Jan 9]. Available from: <https://medium.com/@azimkhan8018/a-beginners-guide-to-deep-learning-with-mnist-dataset-0894f7183344>
7. Training a neural network on MNIST with Keras | TensorFlow Datasets [Internet]. TensorFlow. [cited 2025 Jan 9]. Available from: https://www.tensorflow.org/datasets/keras_example
8. Training a Classifier — PyTorch Tutorials 2.5.0+cu124 documentation [Internet]. [cited 2025 Jan 13]. Available from: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html
9. Kalantar R. How to visualize layer activations in PyTorch [Internet]. Medium. 2024 [cited 2025 Jan 6]. Available from: <https://medium.com/@rekalantar/how-to-visualize-layer-activations-in-pytorch-d0be1076ecc3>
10. Mazur. A Step by Step Backpropagation Example [Internet]. Matt Mazur. 2015 [cited 2025 Jan 13]. Available from: <https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>
11. How to use TensorBoard with PyTorch — PyTorch Tutorials 2.5.0+cu124 documentation [Internet]. [cited 2025 Jan 6]. Available from: https://pytorch.org/tutorials/recipes/recipes/tensorboard_with_pytorch.html
12. Welcome to PyTorch Lightning — PyTorch Lightning 2.5.0.post0 documentation [Internet]. [cited 2025 Jan 6]. Available from: <https://lightning.ai/docs/pytorch/stable/>
13. ALI AWAN A. A Complete Guide to Data Augmentation [Internet]. [cited 2025 Jan 6]. Available from: <https://www.datacamp.com/tutorial/complete-guide-data-augmentation>

14. Kundu R. The Beginner's Guide to Contrastive Learning [Internet]. [cited 2025 Jan 6]. Available from: <https://www.v7labs.com/blog/contrastive-learning-guide>
15. Rubilmax. Introduction au contrastive learning : une forme d'apprentissage auto supervisé [Internet]. Medium. 2022 [cited 2025 Jan 20]. Available from: <https://medium.com/@rubilmax/introduction-au-contrastive-learning-une-forme-dapprentissage-auto-supervis%C3%A9-95c3bc070d1c>
16. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, Krueger G, Sutskever I. Learning Transferable Visual Models From Natural Language Supervision [Internet]. arXiv; 2021 [cited 2025 Jan 9]. Available from: <http://arxiv.org/abs/2103.00020>
17. Hydra | Weights & Biases Documentation [Internet]. [cited 2025 Jan 6]. Available from: <https://docs.wandb.ai/guides/integrations/hydra/>
18. W&B Docs | Weights & Biases Documentation [Internet]. [cited 2025 Jan 6]. Available from: <https://docs.wandb.ai/>
19. Lu S. TheMatrixMaster/mmc-gene [Internet]. 2024 [cited 2025 Jan 8]. Available from: <https://github.com/TheMatrixMaster/mmc-gene>
20. Poklukar P, Vasco M, Yin H, Melo FS, Paiva A, Krägic D. Geometric Multimodal Contrastive Representation Learning [Internet]. arXiv; 2022 [cited 2025 Jan 8]. Available from: <http://arxiv.org/abs/2202.03390>
21. Moshkov N, Becker T, Yang K, Horvath P, Dancik V, Wagner B, Clemont P, Singh S, Carpenter A, Caicedo JC. Predicting compound activity from phenotypic profiles and chemical structures | bioRxiv [Internet]. [cited 2025 Jan 6]. Available from: <https://www.biorxiv.org/content/10.1101/2020.12.15.422887v4.full>
22. Moshkov N, Becker T, Yang K, Horvath P, Dancik V, Wagner BK, Clemons PA, Singh S, Carpenter AE, Caicedo JC. Chemical structures, Cell Painting and transcriptional profiles for compound bioactivity prediction. [Internet]. Zenodo; 2023 [cited 2025 Jan 13]. Available from: <https://zenodo.org/records/7729583>
23. JUMP-Cell Painting Consortium [Internet]. JUMP-Cell Painting Consortium. [cited 2025 Jan 10]. Available from: <https://jump-cellpainting.broadinstitute.org/>
24. Fradkin P, Azadi P, Suri K, Wenkel F, Bashashati A, Sypetkowski M, Beaini D. How Molecules Impact Cells: Unlocking Contrastive PhenoMolecular Retrieval [Internet]. arXiv; 2024 [cited 2025 Feb 3]. Available from: <http://arxiv.org/abs/2409.08302>
25. Chandak P, Huang K, Zitnik M. Building a knowledge graph to enable precision medicine. Sci Data. 2023 Feb 2;10(1):67.
26. Streamlit Docs [Internet]. [cited 2025 Jan 6]. Available from: <https://docs.streamlit.io>
27. What is CRISPR: Your Ultimate Guide [Internet]. Synthego. [cited 2025 Jan 9]. Available from: <https://www.synthego.com/learn/crispr>

28. Chall S. Understanding Molecular Similarity [Internet]. Medium. 2023 [cited 2025 Jan 9]. Available from: <https://medium.com/@santuchal/understanding-molecular-similarity-51e8ebb38886>
29. rdkit/rdkit [Internet]. RDKit; 2025 [cited 2025 Jan 20]. Available from: <https://github.com/rdkit/rdkit>
30. Neo4j documentation - Neo4j Documentation [Internet]. Neo4j Graph Data Platform. [cited 2025 Jan 6]. Available from: <https://neo4j.com/docs/docs/>
31. Using a Knowledge Graph to Implement a RAG Application [Internet]. [cited 2025 Jan 6]. Available from: <https://www.datacamp.com/tutorial/knowledge-graph-rag>
32. Sepasdar Z, Gautam S, Midoglu C, Riegler MA, Halvorsen P. Enhancing Structured-Data Retrieval with GraphRAG: Soccer Data Case Study [Internet]. arXiv; 2024 [cited 2025 Jan 6]. Available from: <http://arxiv.org/abs/2409.17580>
33. Nova M. RETRIEVAL AUGMENTED GENERATION (RAG) WITH LANGCHAIN. 2024 Dec 18;61.
34. GenAI Ecosystem - Developer Guides [Internet]. Neo4j Graph Data Platform. [cited 2025 Jan 9]. Available from: <https://neo4j.com/developer/genai-ecosystem/>
35. sentence-transformers/all-MiniLM-L6-v2 · Hugging Face [Internet]. 2024 [cited 2025 Jan 6]. Available from: <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>
36. How does `GraphCypherQACChain.from_llm` construct answers based on a cypher query's results? · langchain-ai/langchain · Discussion #15313 [Internet]. GitHub. [cited 2025 Jan 9]. Available from: <https://github.com/langchain-ai/langchain/discussions/15313>
37. Klose C. ChrisDelClea/streamlit-a graph [Internet]. 2025 [cited 2025 Jan 10]. Available from: <https://github.com/ChrisDelClea/streamlit-a graph>

VI. Table des figures

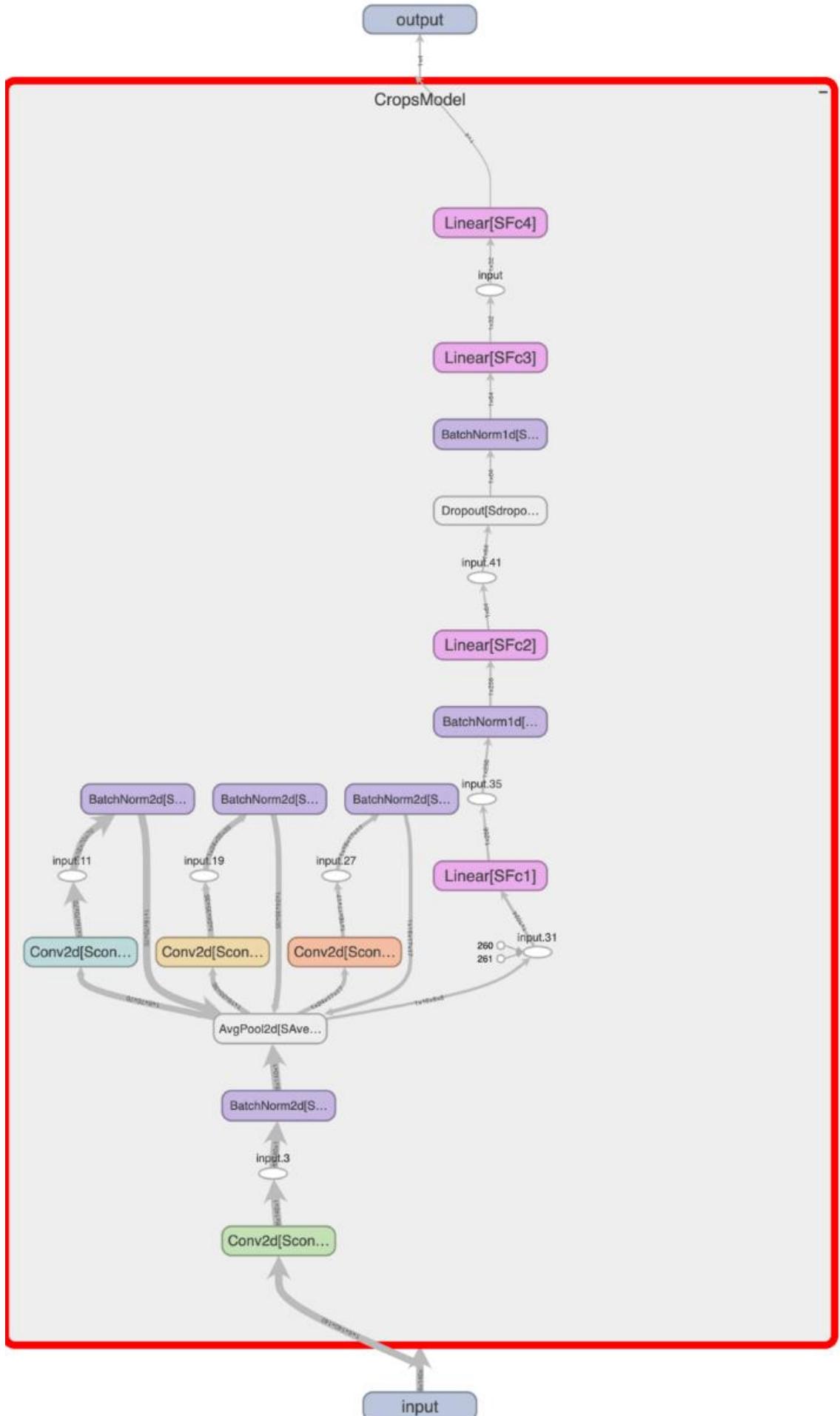
Figure 1: Organigramme de Ksilink	2
Figure 2: Recto de mon brouillon de calcul de back propagation	9
Figure 3: Validation Loss du model dans les premiers essais.....	10
Figure 4: Validation accuracy du model dans les premiers essais	10
Figure 5: Validation accuracy du model après avoir modifié le nombre de répétitions d'entrainements	11
Figure 6: Graphique du model via tensorboard (Annexe 1 pour avoir une version pleine page).....	13
Figure 7: Matrice de confusion à l'étape 1 où chaque colonne du tableau contient une classe prédite par l'algorithme et les lignes des classes réelles. Les valeurs sur la diagonale représentent les prédictions correctes alors que les valeurs en dehors de la diagonale sont celle incorrectement prédites par le modèle.	14
Figure 8: Affichage des quatre canaux d'une image mal classifiée	14
Figure 9: Gradient descent et affichage des prédictions du modèle	15
Figure 10: Visualisation selon la couche convulsive	16
Figure 11 : Validation accuracy final	17
Figure 12: Validation loss final	17
Figure 13: Représentation des deux espaces latents obtenue, provenant de l'article scientifique[3]	19
Figure 14:Représentation du générateur de molécules grâce à une morphologie cible, récupérée dans l'article scientifique[3]. La morphologie cible est d'abord transformé en un vecteur et la similarité entre ce vecteur et les vecteurs des molécules générées est utilisé comme récompense pour le model.	20
Figure 15 : Explication par image du multimodal contrastive learning tirée de l'article scientifique de CLIP (15).....	21
Figure 16: Résultats obtenu en interne avec les données de l'article se rapprochant des résultats de l'étude (en utilisant leurs datas)	23
Figure 17:Résultat de la publication[3]	23
Figure 18 : Graphe de connaissances de Ksilink, les relations en couleur sont les relations que j'ai ajoutées.....	26
Figure 19: Principe du Graph RAG[34]	27
Figure 20: Exemple de passage entre connaissance général et graphe de connaissances	29
Figure 21: Exemple d'échange avec mémoire.....	30
Figure 22: Problème avec le model qui ne veux pas donner les réponses obtenues. On peut observer sur la partie de droite la requête effectuer par le modèle ainsi que les résultats obtenus sous Full Context. Pourtant le modèle répond qu'il ne connaît pas la réponse.....	31
Figure 23 : Première réponse du model, obtenue en demandant des explications au modèle de pourquoi il n'a pas fourni de réponse.	31
Figure 24 : Seconde réponse du model, obtenue en demandant au modèle pourquoi il n'a pas répondu malgré le fait qu'il ait les réponses.	31

Figure 25: Exemple d'un graphe généré à partir d'une question posée au chatbot. Dans cet exemple, les nœuds sont filtrés et organisés pour refléter uniquement les informations pertinentes, avec une mise en évidence du nœud spécifié dans la question.....	33
Figure 26: Ajout de la barre de recherche et du guide de questions	35
Figure 27: Mode développeur activé, permettant ainsi de choisir le modèle, le nombre de réponses, de choisir la recherche exacte et de voir le prompt et la requête cypher. 37	
Figure 28: Expliquer des graphes imposant rapidement avec un Large language model.	38

VII. Annexes

i. Annexe 1 :

Schéma de mon modèle via Tensorboard, comprenant plusieurs couches de convolution, de la batch normalisation, de l'average pooling, un dropout, ... Le modèle prend en entrée une image avec 4 canaux et renvoie en sortie une prédiction entre les quatre classes.



ii. Annexe 2 :

Entrainement de manipulation de fichier .tar. J'ai alors dû créer une page Streamlit capable soit de charger des données en provenance d'un fichier .tar puis de les charger dans un fichier pickle pour réutilisation rapide, soit de charger directement un fichier pickle déjà créé.

DinoV2 data processing and visualization

How would you like to load the data?

Load from pickle
 Process tar files

Chose a pickle file to load

/mnt/shares/L/CHEMINFO/Leon/Streamlit_pickles/CP1_SC1-01-25_DinoV2.pkl

Load Pickle Data

Data loaded from pickle!

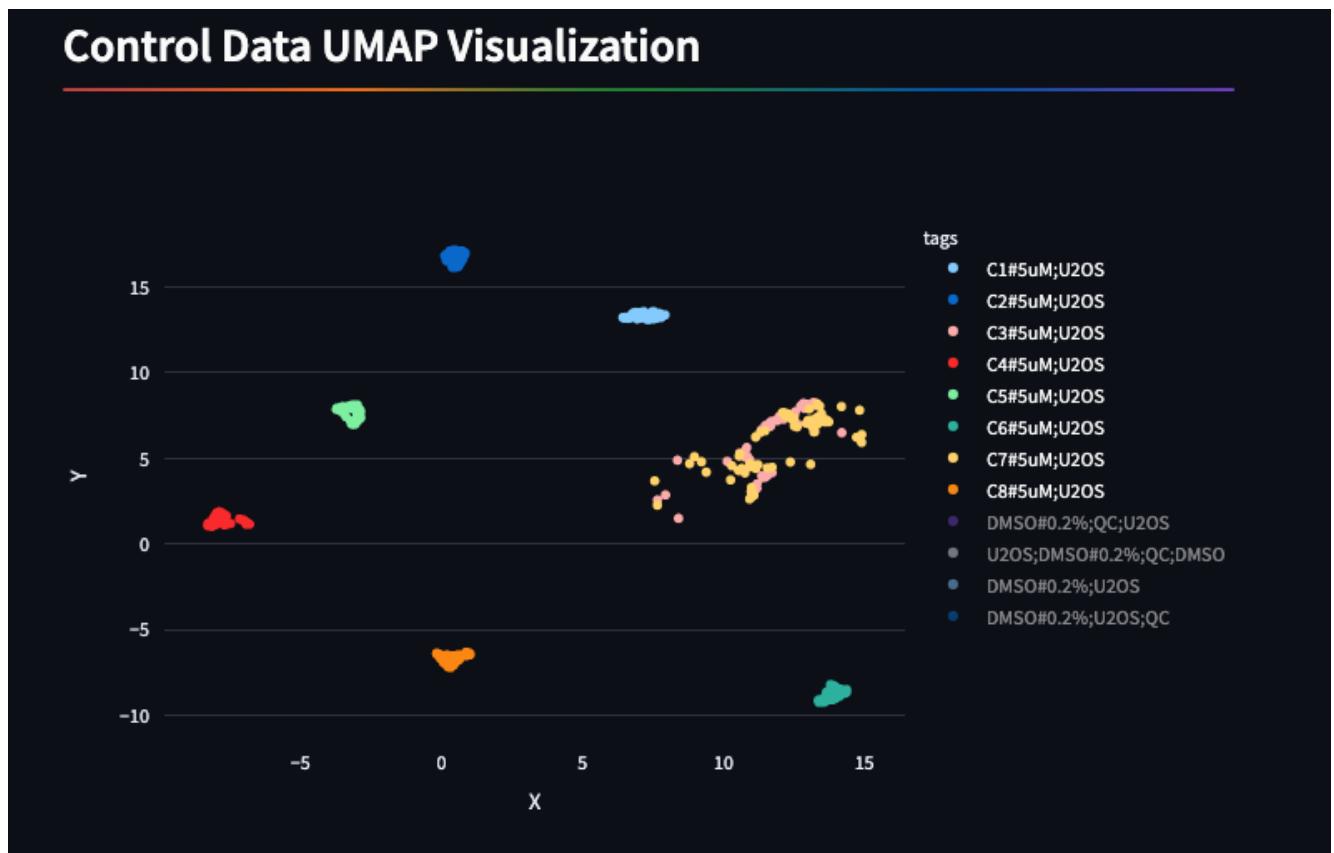
Control Data and Visualization

Control Data

	Feature_0	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5	Feature_6	Feature_7	Feature
0	0.0099	-2.9309	0.0585	1.2959	1.9823	-4.8644	-0.7131	-1.4634	0.20
1	-0.1743	-2.6437	0.1572	1.2422	1.6195	-4.9182	-1.0237	-1.4198	0.0
2	-0.1999	-2.8277	0.103	1.2463	1.957	-4.8927	-0.7282	-1.5628	0.23
3	0.2303	-3.2426	0.2873	1.1115	1.8486	-4.7632	-0.7506	-1.4496	0.20
4	-0.2739	-2.8223	0.0195	1.3934	2.0604	-4.943	-0.743	-1.4325	0.18
5	-0.3052	-2.948	0.0672	1.2539	2.015	-4.9629	-0.8099	-1.4934	0.22
6	-0.1004	-2.8891	0.1183	1.2655	1.9954	-4.8965	-0.7674	-1.5534	0.2
7	0.15	-3.0548	0.3849	0.8852	1.6594	-4.8264	-0.8858	-1.6027	0.34

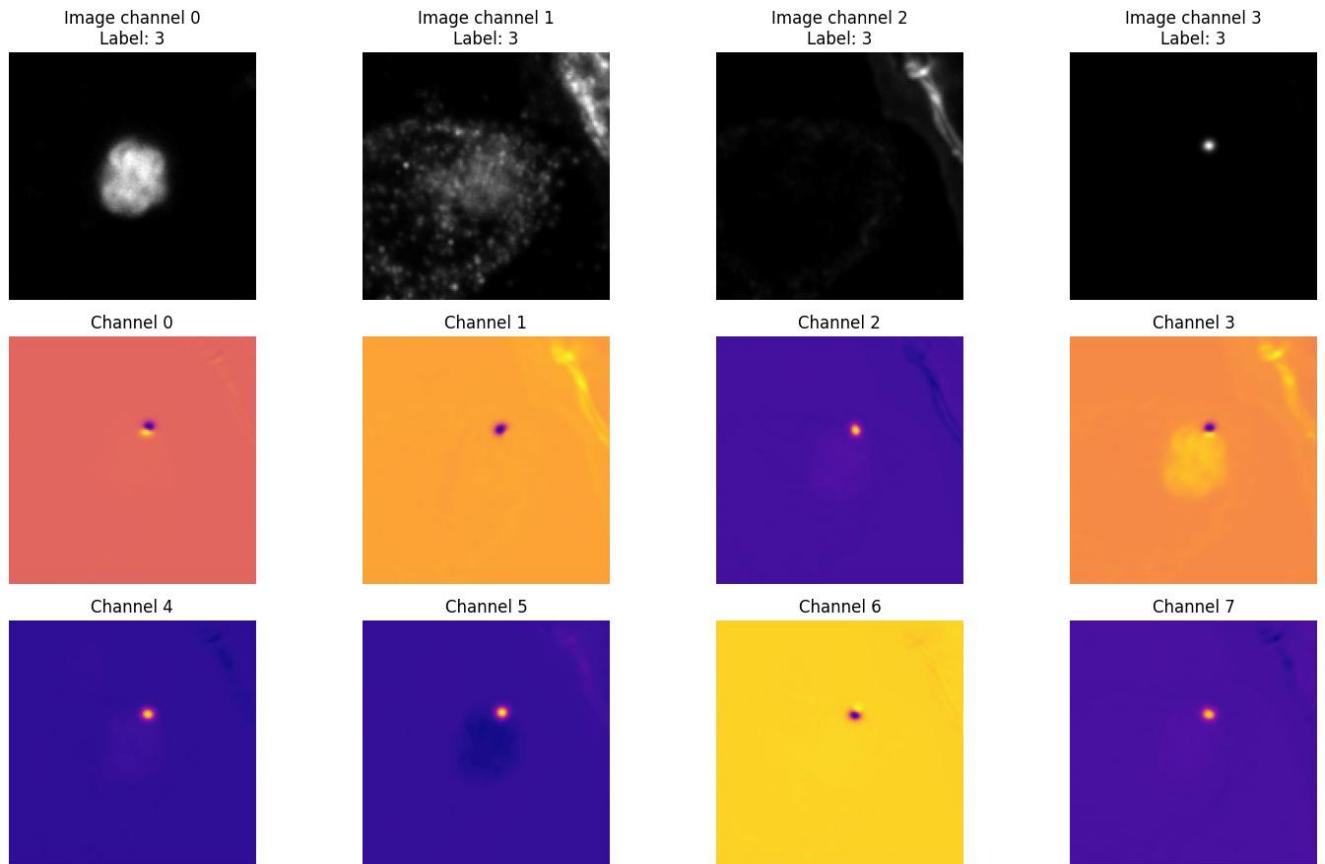
iii. Annexe 3 :

Une fois les données chargées, création et affichage interactif d'un UMAP permettant de voir rapidement si les différentes catégories de données sont bien séparées.



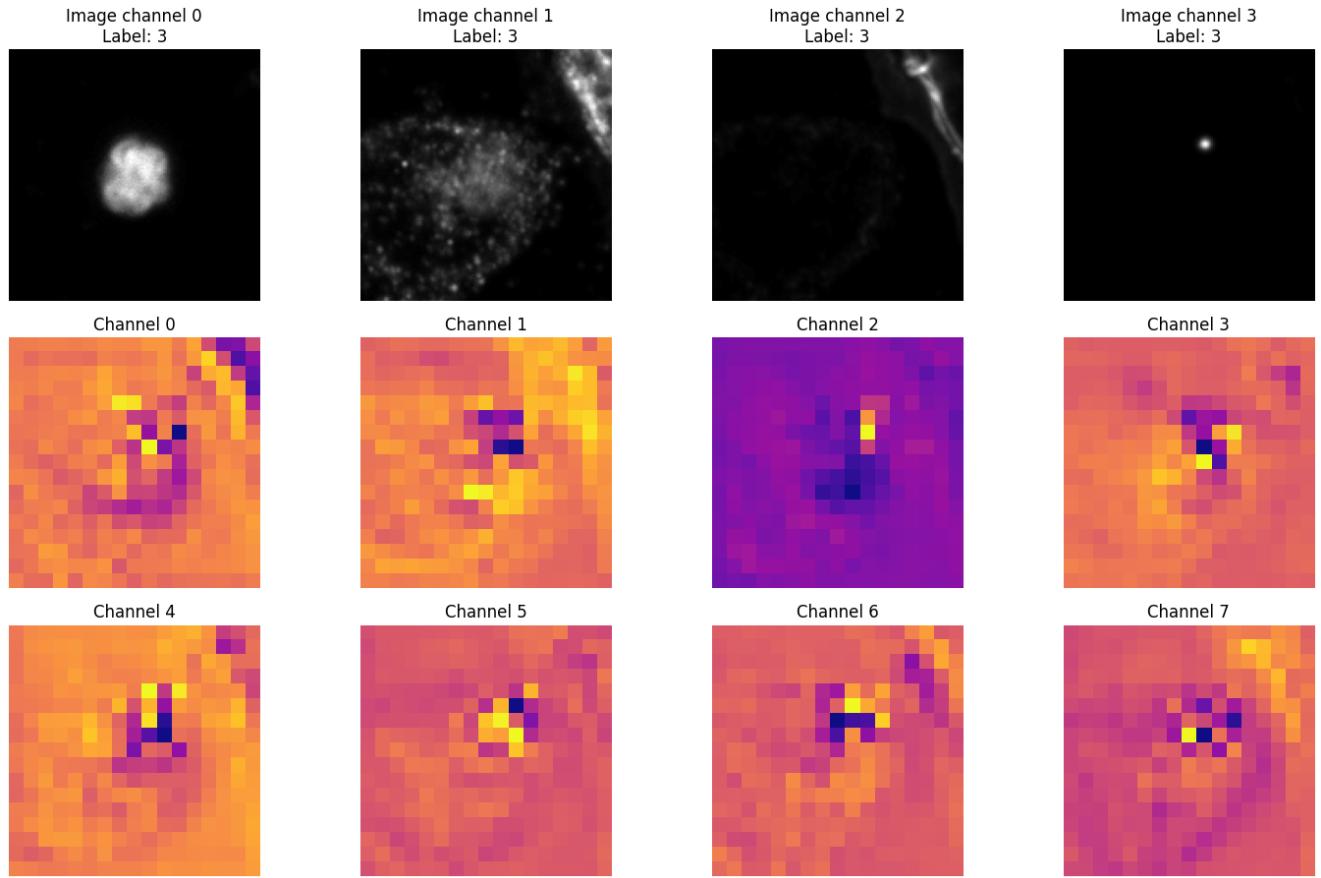
iv. Annexe 4 :

Images des huit canaux de la première couche convulsive ainsi que des quatre canaux originaux. Dans ce gradient de couleur, plus c'est clair, plus c'est intense, et plus c'est foncé, moins c'est intense. La première couche convulsive prend 4 canaux en entrée et en ressort huit.



v. Annexe 5 :

Image des huit premiers canaux de la quatrième couche convulsive ainsi que des quatre canaux originaux. Dans ce gradient de couleur, plus c'est clair, plus c'est intense, et plus c'est foncé, moins c'est intense. La quatrième couche convulsive prend 24 canaux en entrée et en ressort 16, dont voici les 8 premiers.

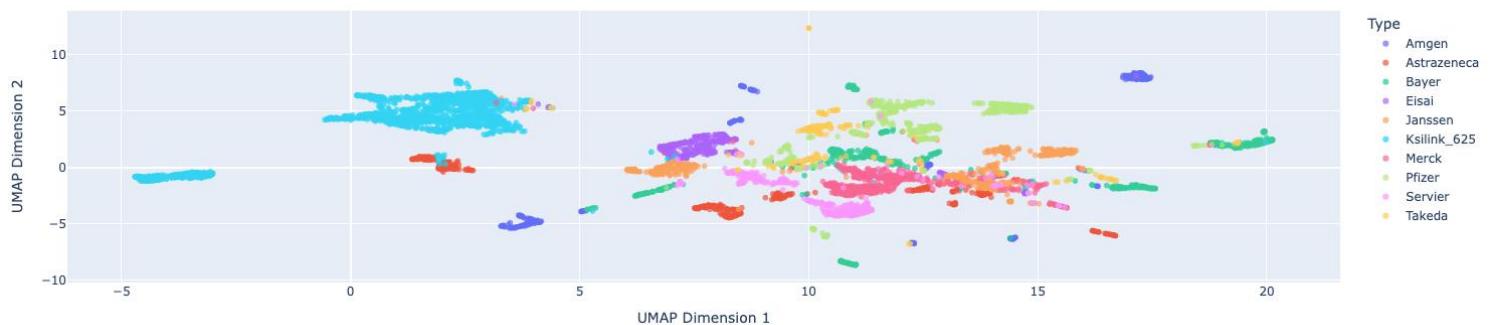


vi. Annexe 6 :

Graphique représentant les données provenant de différentes sources. Chaque source ayant pour la plupart des composés différents sauf un solvant commun à toutes les sources, le DMSO. Le but est que la source (type de microscope, date, température, ...) n'influe pas sur les données. Il faut donc recalibrer les données pour faire en sorte que tous les DMSO soient groupés, tout en faisant en sorte d'avoir malgré tout, des groupes de composés distincts.

Ici on peut voir que les composés sont groupés par source et on ne voit pas les DMSO groupés, il faut donc corriger cela.

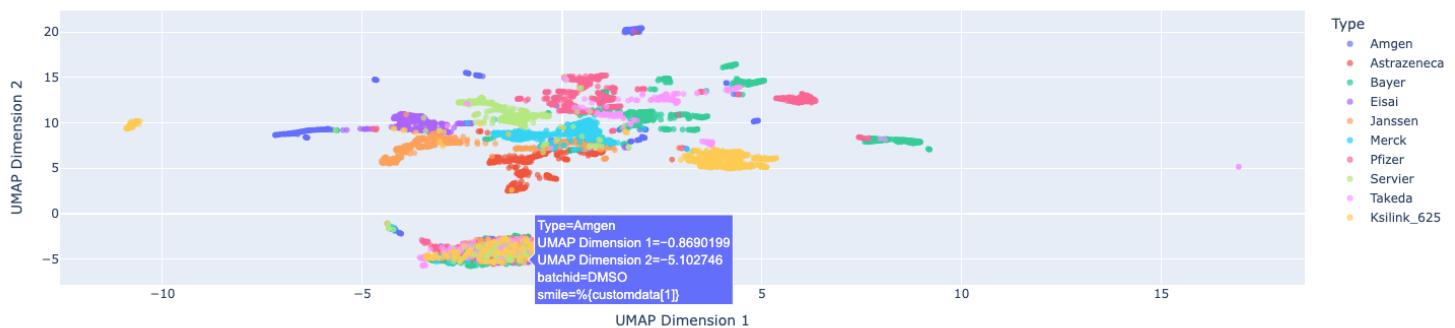
UMAP after cleaning



vii. Annexe 7 :

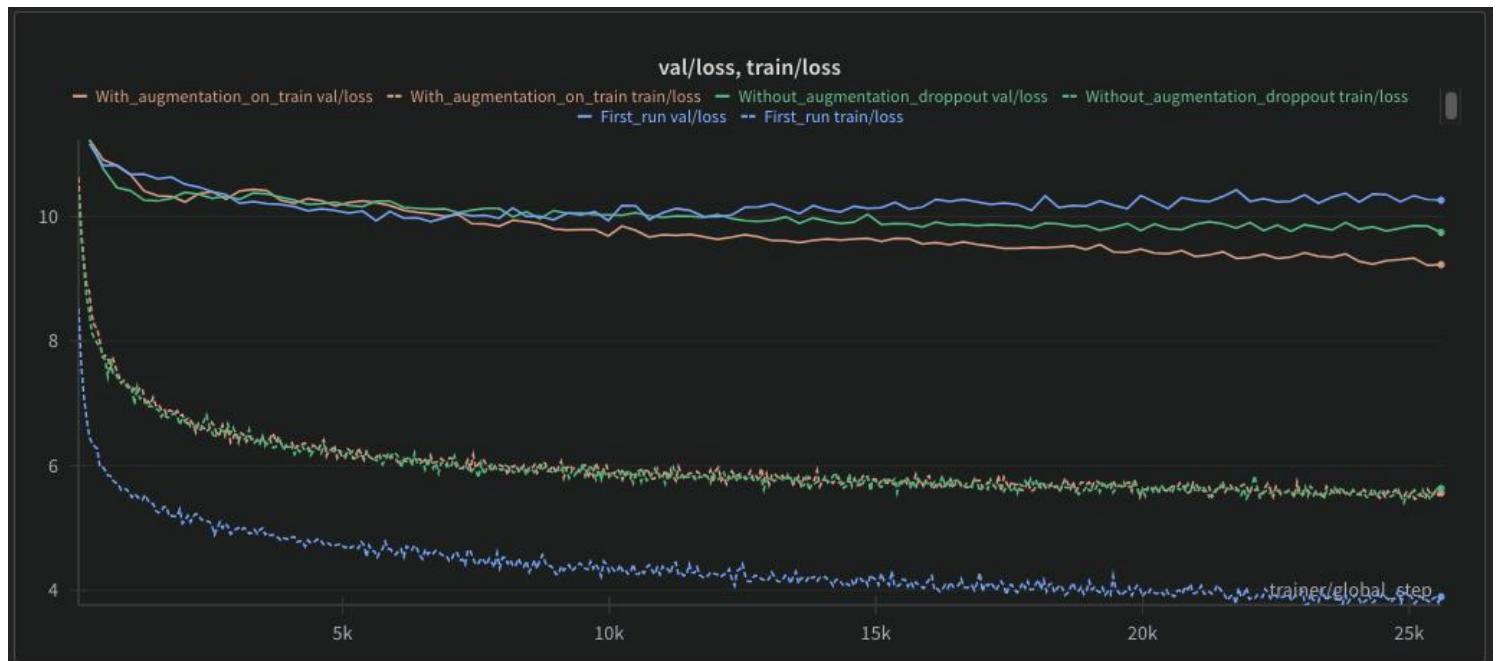
Ici, on peut voir que les composés sont légèrement groupés par source, mais que tous les DMSO sont groupés au même endroit. La correction a donc bien été effectuée.

UMAP after just Combat



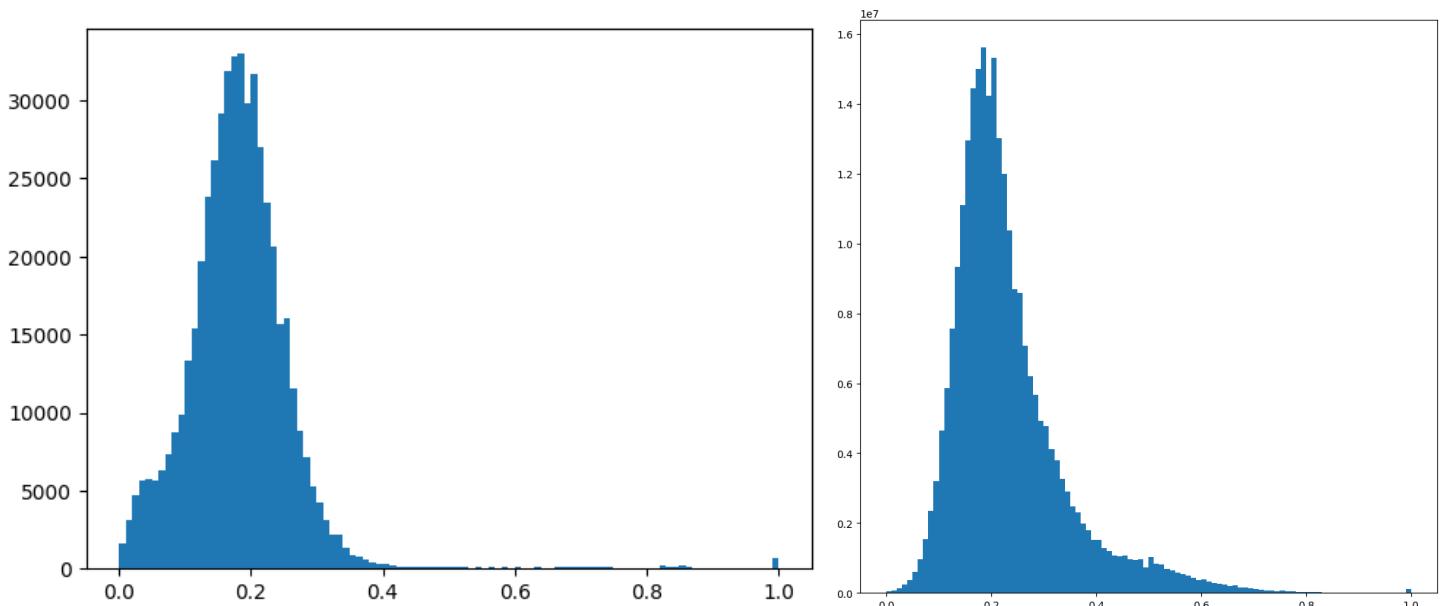
viii. Annexe 8 :

Résultats obtenus avec plusieurs tentatives de jeux de données, en ligne pleine la courbe de perte pour le jeu de validation et en ligne pointillée celle obtenue avec le jeu d'entraînement. On peut voir une différence significative entre l'entraînement et la validation, ce qui montre bien que le modèle sur-apprend des données d'entraînement et n'arrive pas à généraliser aux données de validation.



ix. Annexe 9 :

Sur ces deux images, on peut voir un comparatif de la similarité de Tanimoto pour les données de Ksilink (à gauche) et les données de la publication (à droite). Ici, les données de Ksilink ont été sélectionnées pour être les plus similaires possible. Pourtant, malgré cela, les données de Ksilink ne sont pas aussi similaires que le sont les données de la publication, et cela malgré le fait que les données de Ksilink contiennent seulement 700 composés contre 16170 pour les données de la publication.



MORALES Léon

Rapport de stage ST40 - P2024

Développement d'un Large Language Model au service de l'industrie pharmaceutique

Le sujet de mon stage portait sur le développement d'un **Large Language Model (LLM)** appliqué à l'industrie pharmaceutique, en exploitant les données internes de Ksilink. Mon stage a été réparti en deux grands projets :

Construction d'un espace latent commun : Cette étape visait à intégrer plusieurs types de données, notamment des descriptions moléculaires (en format SMILES) et des informations extraites d'images de cellules. L'objectif était de représenter ces données dans un espace vectoriel partagé, facilitant ainsi leur analyse et leur exploitation conjointe dans le but de créer un générateur de molécules chimiques.

Graphe de connaissances et lien avec le LLM : Cette partie avait pour but d'enrichir le graphe de connaissances en ajoutant de nouvelles relations et informations. Cela incluait également l'amélioration de l'interface reliant le graphe de connaissances à un LLM, pour permettre des interactions en langage naturel et l'ajout de fonctionnalités rendant l'outil plus instinctif et compréhensible pour les biologistes.

KSILINK
16 rue d'Ankara
67 000 Strasbourg
www.ksilink.com