EPITA

Spécialisation SCIA-G
(Data Science & Intelligence Artificielle)

Neuro-Graph Trader

Architecture Multi-Agents et Graph Transformers Dynamiques pour l'Optimisation Séquentielle

Étudiant :
Wilfried BEMELINGUE

Dominantes Techniques :
Dynamic GNN · Residual RL · Stochastic Modeling · XAI

Année Académique 2025-2026

Table des matières

1 Genèse du Projet et Objectifs

Résumé en une phrase : Ce projet vise à concevoir une intelligence artificielle capable de gérer automatiquement un portefeuille d'actions en apprenant les relations dynamiques entre entreprises sous forme de graphe et en optimisant des décisions d'investissement par apprentissage par renforcement.

1.1 L'Inspiration : Stanford CS224W

Ce projet tire son origine de l'étude approfondie de l'article *"Stock Market Forecasting with Differential Graph Transformer"* (Stanford CS224W / Medium) [1]. Cet article démontre la pertinence des structures de graphes pour modéliser les dépendances non-euclidiennes des marchés. Cependant, il se limite à une tâche de prédiction supervisée (Forecasting).

1.2 Objectif : De la Prédiction à la Décision "From Scratch"

Mon ambition est de ré-implémenter cette approche *from scratch* pour maîtriser l'intégralité de la chaîne de valeur, et de l'étendre considérablement. L'objectif est de passer d'un modèle prédictif à un **Agent Autonome Décisionnel** capable d'optimiser une politique complexe via le Reinforcement Learning, le tout intégré dans une interface de visualisation interactive.

2 Problématique Machine Learning

Le problème traité est algorithmique et non purement financier : **Prise de décision automatique sur un marché financier représenté comme un graphe évolutif**.

Comment entraîner une IA à converger vers une politique optimale (π^*) quand :

1. Les données d'entrée sont des graphes dynamiques $G_t(V, E_t)$ dont la topologie change à chaque pas de temps t.
2. L'environnement est stochastique avec un très faible ratio signal/bruit.

3 Pipeline d'Ingénierie des Données : De l'Extraction au Graphe

La qualité de la représentation graphale conditionne la performance du modèle. Nous avons conçu un pipeline ETL strict pour convertir des flux financiers bruyants en tenseurs structurés pour PyTorch Geometric.

3.1 Extraction et Nettoyage (Raw Data)

— **Source :** Extraction via l'API `yfinance` des données OHLCV (Open, High, Low, Close, Volume) pour les constituants du S&P 500.

1. https://medium.com/stanford-cs224w/stock-market-forecasting-with-differential-graph-transformer

— **Nettoyage :** Gestion des valeurs manquantes (Forward Fill) et alignement des dates.

— **Stationnarité (EDA) :** Les GNN convergent mal sur des données non-stationnaires. Nous transformons les prix bruts en *Log-Returns* :

$$r_t = \ln(P_t) - \ln(P_{t-1})$$

Un test de Dickey-Fuller Augmenté (ADF) est appliqué pour valider la stationnarité des séries en entrée.

3.2 Feature Engineering des Nœuds (Node Embeddings)

Chaque entreprise est un nœud $v \in V$. À l'instant t, ce nœud n'est pas représenté par une seule valeur, mais par un vecteur de caractéristiques $x_v^{(t)}$ (Feature Vector) de dimension D. Ce vecteur inclut :

— Les rendements passés (Lagged returns).

— Des indicateurs techniques calculés (RSI, MACD, Bandes de Bollinger) pour capturer le momentum et la volatilité locale.

— **Résultat :** Une matrice de Features $X_t \in \mathbb{R}^{N \times D}$.

3.3 Enrichissement NLP des Nœuds (Signal Informationnel Exogène)

En complément des caractéristiques purement financières, chaque nœud du graphe peut être enrichi par des signaux informationnels issus du traitement automatique du langage naturel (NLP), afin de capturer le contexte exogène du marché non immédiatement observable dans les prix.

Ces signaux peuvent inclure :

— un score de sentiment agrégé à partir de news financières ou de communiqués,

— le volume d'articles négatifs ou anormalement fréquents concernant une entreprise ou un secteur,

— une mesure de la volatilité du sentiment informationnel.

Ces variables sont intégrées au vecteur de caractéristiques du nœud sous la forme :

$$x_v^{(t)} = [x_v^{(t,\text{finance})} \,||\, x_v^{(t,\text{NLP})}]$$

Le NLP est considéré comme un **signal exogène**, n'influençant pas directement la dynamique du marché, mais enrichissant l'état observé par l'agent. Selon la disponibilité des données, ces signaux peuvent provenir de sources réelles, de proxies agrégés ou de simulations contrôlées.

3.4 Apprentissage de la Topologie (Graph Construction)

C'est l'étape critique. Contrairement aux réseaux sociaux, le graphe financier n'est pas explicite. Nous devons inferer les arêtes (*Edges*) dynamiquement.

Méthode par Fenêtre Glissante (Rolling Window) : Pour chaque pas de temps t, nous considérons une fenêtre historique W (ex : les 30 derniers jours).

1. **Calcul des Corrélations :** Nous calculons la matrice de corrélation de Pearson par paires entre tous les actifs sur la fenêtre W.
2. **Création des Arêtes (Thresholding) :** Pour éviter un graphe complet (dense) qui provoquerait de l'*Oversmoothing* (lissage excessif du signal dans le GNN), nous appliquons un seuillage strict :

$$A_{ij}^{(t)} = \begin{cases} \rho_{ij} & \text{si } |\rho_{ij}| > \theta \text{ (ex : 0.6)} \\ 0 & \text{sinon} \end{cases}$$

3. **Dynamisme :** Cette matrice d'adjacence A_t est recalculée à chaque étape. Le modèle voit donc des liens se créer (ex : le secteur Tech se synchronise) ou se rompre.

3.5 4. Tensorisation (PyTorch Geometric)

Le pipeline final encapsule ces données dans des objets `Data` de PyTorch Geometric contenant :

— `x` : La matrice des features des nœuds ($N \times D$).
— `edge_index` : La liste des connexions (format COO).
— `edge_attr` : Le poids des corrélations (pour que le GNN sache si le lien est fort ou faible).

4 Architecture Méthodologique : Dual-Agent Residual RL

Pour pallier l'instabilité bien connue du RL sur des données bruitées, nous proposons une architecture collaborative à deux agents.

4.1 Agent B : Le "Tuteur" (Baseline - Buy & Hold)

Agent passif et déterministe représentant la moyenne du marché. En termes ML, il agit comme un *biais inductif* fort, fournissant une politique de base robuste.

4.2 Agent A : L' "Élève" (Active - Neuro-Graph Model)

C'est notre modèle Deep Learning (Dynamic Graph Transformer + LSTM). Il n'apprend pas à gérer le portefeuille de zéro (*Tabula Rasa*), mais utilise l'**Apprentissage Résiduel**.

4.3 Orchestration (Residual Policy)

L'action finale est une combinaison linéaire :

$$Action_t = \underbrace{W_{\text{Tuteur}}}_{\text{Baseline}} + \alpha \cdot \underbrace{\text{Policy}_\theta(G_t)}_{\text{Correction IA}} \tag{1}$$

L'Agent A apprend donc uniquement à calculer les **ajustements optimaux** (ΔW) pour sur-performer le tuteur. L'entraînement est réalisé dans l'environnement **FinRL** (OpenAI Gym Wrapper), dans lequel nous injectons cette architecture customisée.

5 Traduction des Concepts : Finance → ML

Cette section établit un pont explicite entre les concepts financiers classiques et leur formulation en Machine Learning, afin de clarifier l'interdisciplinarité du projet. Pour aborder ce projet avec un regard de Data Scientist :

— **Le Benchmark → Teacher Forcing :** L'agent Buy & Hold sert de professeur pour guider les gradients au début de l'apprentissage.

— **Ratio de Sharpe → Loss Régularisée :** Maximiser le rendement ajusté au risque équivaut à maximiser une récompense avec pénalité de variance : $R = \mu - \lambda\sigma^2$.

— **Turbulence → Détection Out-of-Distribution (OOD) :** Nous utilisons la distance de Mahalanobis pour détecter les changements de régime statistique (Distribution Shift) et adapter dynamiquement le taux d'exploration/exploitation.

— **Information Textuelle → Signal Exogène de Risque :** Les signaux issus du NLP sont utilisés pour caractériser le contexte informationnel du marché (stress médiatique, annonces négatives, incertitude), et peuvent intervenir dans la modulation dynamique de l'aversion au risque de l'agent, sans modifier la dynamique sous-jacente de l'environnement.

6 Valeur Scientifique et Analyse

La contribution recherche se focalisera sur trois axes :

1. **Benchmark Architectural :** Comparaison quantitative (Convergence, Sharpe) entre une architecture *Dynamic Graph Transformer* (DGT) et une architecture statique (GAT + LSTM).
2. **Ablation Study :** Analyse de l'impact du *Residual Learning* sur la stabilité des gradients par rapport à un agent PPO standard.
3. **Explainable AI (XAI) :** Analyse des cartes de chaleur des *Attention Weights* pour vérifier si le modèle "comprend" les relations de contagion sectorielle.
4. **Analyse de l'Impact des Signaux Informationnels :** Étude de l'apport des signaux NLP en tant qu'information exogène, via des expériences d'ablation comparant les performances du modèle avec et sans enrichissement informationnel.

7 Périmètre Réaliste et Livrables

Afin de garantir la réussite du projet M2 :

7.1 Gestion du Risque (Fallback Strategy)

Si l'entraînement du DGT complet s'avère trop instable, le projet basculera sur une architecture **GAT Statique** (recalculée périodiquement) qui est plus robuste, assurant ainsi un résultat exploitable pour la soutenance.

7.2 Livrables Finaux

— **Code Source :** Pipeline ETL + Agents RL entraînés (PyTorch/FinRL).

— **Démonstrateur Streamlit :** Application Web interactive permettant de visualiser le graphe dynamique en temps réel, les courbes de performance comparées (Agent A vs B) et les poids d'attention (XAI).

— **Rapport Scientifique :** Analyse des résultats et des limites.

8 Conclusion

Ce projet **Neuro-Graph Trader** est un défi d'ingénierie ML complet. Il combine la complexité des GNN dynamiques, l'instabilité du RL, et une architecture multi-agents résiduelle pour prouver qu'une IA peut extraire de la valeur de la structure relationnelle des données, au-delà des simples séries temporelles.