Analyse et Prédiction des Evolutions de Titres Boursiers avec Transformers

Louis Caron
Département de Génie des Systèmes
Ecole de Technologie Supérieure
Montréal, Canada
Email: louis.caron.5@ens.etsmtl.ca

David Chakroun
Département de Génie des Systèmes
Ecole de Technologie Supérieure
Montréal, Canada
Email: david.chakroun.1@ens.etsmtl.ca

***Abstract*—La majorité des modèles utilisés pour de la prédiction financière ne peuvent actuellement traiter qu'une séquence (un titre boursier) à la fois. Nous supposons que cette approche limite le potentiel des modèles en masquant les relations entre les différents titres.**
En nous basant sur les progrès réalisés en NLP par les Transformers [1] et BERT [2], nous proposons un modèle capable de traiter différents titres simultanément pour mieux modéliser leurs relations et la complexité du marché. Notre modèle est constitué de plusieurs encoders superposés et est conceptuellement très similaire à BERT [2]. Il est cependant adapté pour pouvoir traiter plusieurs séquences simultanément et prendre en compte leurs les relations à travers le temps grâce au mécanisme d'Attention, qui permet de mettre en relation les items contenu dans une séquence.
Notre modèle est entraîné avec une double loss, pour classifier l'évolution des titres mais aussi pour prédire précisément leurs valeurs futures. Malgré des résultats préliminaires assez mauvais, notre modèle ouvre la voie à une nouvelle famille de modèles pour l'analyse financière. Nous montrons que le mécanisme d'Attention peut être appliqué à des séquences boursières, mais que cela nécessite des précautions pour limiter son coût en mémoire.
Afin d'entraîner notre modèle, nous avons aussi généré un dataset permettant de comparer différents titres boursiers. Notre dataset contient les informations journalières standardisées de la plupart des titres du S&P500, ainsi que des devises, des commodités et d'autres index sur une durée de plus de 15 ans. Ce dataset peut être mis à jour automatiquement pour l'intervalle de temps voulu.

1. Introduction

Depuis l'invention de la finance de marché, la prédiction des cours boursiers a toujours été une activité ayant un potentiel lucratif élevée et prisée par de nombreuses organisations financières. Les dernières années ont particulièrement étaient témoins de grandes évolutions dans le domaine de l'ingénierie financière. Ces innovations ont eu lieu parallèlement à la démocratisation des techniques de Deep Learning. Les grands progrès en Deep Learning, et notamment en traitement de langage naturel et de série temporelle, ont en effet permis la création de nouvelles générations de modèles financiers plus complexes que ceux existants jusqu'ici. De nouveaux modèles financiers continuent d'être régulièrement créés au fur et à mesure des avancés en Deep Learning afin d'améliorer les prédictions de marché [3].

Cependant, la plupart de ces modèles financiers n'étudient qu'un nombre réduit de cours boursiers à la fois [3, 5, 6] sans prendre en compte toutes les interactions complexes entre les différents cours. Nous admettons comme axiome que cela réduit la précision des modèles [3, 6], et que les cours boursiers sont tous reliés avec des interactions complexes. Il est admis que le marché se comporte comme un système complexe, avec une forte non-linéarité qui le rend compliqué à analyser. Les cours boursiers régissent par exemple à de nombreux facteurs extérieurs communs, comme la 'santé' de l'économie d'un pays, mais leurs réactions face à ces facteurs ne sont pas égales.

En se basant sur les récents progrès fait en NLP et notamment avec les Transformers [1, 2], nous supposons qu'il est possible de mieux modéliser la complexité du marché et les relations entre les différents cours. Les Transformers utilisent un mécanisme d'Attention [1] qui permet de modéliser les relations entre tous les composants d'une séquence. Cependant, les modèles utilisant des Transformers sont faits pour traiter une séquence de texte 1D et ils ne peuvent pas directement être utilisés sur des données financières qui contiennent plusieurs séquences.

Dans cet article, nous souhaitons présenter un modèle utilisant un mécanisme d'Attention adapté pour des données financières. Ce modèle est basé sur l'architecture du modèle state-of-the-art BERT [2] et est capable d'analyser et traiter plusieurs séquences financière en même temps. Nous générons aussi notre propre dataset contenant les informations de plusieurs titres sur une durée de plus de 15 ans afin de pouvoir entraîner notre modèle. Vu la complexité du modèle et nos ressources limités, nous n'avons pas pour but d'obtenir des résultats surpassant ceux des modèles actuels. Nous souhaitons seulement présenter un aperçu, une preuve de concept, que les modèles utilisés en NLP et basés sur l'Attention peuvent être utilisés sur des séries financières.

2. Contexte

Dans cette section, nous présentons le contexte financier de notre projet ainsi que ses liens avec les récents progrès en Deep Learning, et particulièrement en traitement de séquence.

2.1. Contexte Financier

Un marché financier a pour but de permettre la vente et achat de différents titres financiers. Un titre est un actif qui se vend et s'achète en bourse. Parmi ces titres, on retrouve des actions (shares) qui sont des parts d'entreprises, ainsi que des contrats portant sur des monnaies ou des matières premières (pétrole, blé, cuivre...) ou encore des produits plus évolués comme des ETF. Une bourse est un espace virtuel dans lequel il est possible d'effectuer des transactions avec des titres. Il existe des bourses spécialisés dans certains produits financiers comme les actions, les matières premières (Chicago), les monnaies (FOREX) et des indexes (produits financiers basés sur plusieurs titres).

Les prix des titres suivent la loi de l'offre et de la demande, ils évoluent donc dans le temps. La spéculation est une activité qui vise à générer un gain en pariant sur les évolutions des prix des titres. Le but pour un spéculateur est d'acheter au prix bas et revendre au prix haut. Cette activité comporte cependant un certain niveau de risque. Prédire l'évolution des marchés est compliqué, car les marchés sont des systèmes complexes qui s'adaptent très rapidement à de nouvelles informations.

Avec l'informatisation croissante de la finance, l'évolution du cours d'un titre peut être représenté comme une série temporelle, une séquence, contenant le prix du titre (et d'autres informations) à différent instants. Malgré son informatisation, la finance reste un milieu assez opaque il existe assez peu d'information disponibles gratuitement en ligne, surtout en ce qui concerne les données historiques. La plupart des informations sont détenues par des institutions financières et sont donc payantes. Les données publiques sont non standardisées et contiennent des valeurs manquantes.

Compte tenu des gains potentiels, les institutions financières essayent de modéliser les marchés pour prédire leurs évolutions. De nombreux modèles sont ainsi régulièrement inventés, s'inspirant des dernières innovations en informatique et intelligence artificielle [3, 5]. Avec l'avènement du machine learning, de nouveaux types de modèles ont vu le jour, comme des modèles statistiques tels que des Random Forest ou des Régressions Logistiques. Plus récemment et avec la démocratisation du Deep Learning, les réseaux récurrents (RNN) et LSTM [3, 5, 6] sont souvent utilisés.

Les modèles actuels sont, pour la majorité, seulement conçus pour traiter une séquence, un titre à la fois. Or, la plupart des titres financiers ont des relations entre eux. Les titres forment en effet un système complexe et les relations entre chaque titres sont très mal définies [3]. Par exemple, le cours d'une compagnie aérienne dépend du cours du pétrole et de celui de ses concurrents. Vu que ces relations entre titres sont mal définies et complexes, on ne peut pas "prouver" l'existence de ces liens, on considère leur existence comme axiome. On peut facilement voir que les cours boursiers sont dépendants de certains facteurs extérieurs communs, comme les conditions économiques dans un certain pays ou certaines matières premières. Encore une fois, le marché se comporte comme un système complexe en constante adaptation, ce qui le rend difficile à modéliser.

2.2. Transformers et Attention

Pendant longtemps, les réseaux récurrents et leurs variantes ont été considéré comme les meilleurs modèles de traitement de texte. Cela changea avec l'invention des Transformers, présentés pour la premières fois en 2017 dans l'article *Attention Is All You Need* [1]. Un Transformer est une unité neuronal complexe utilisé initialement pour de la traduction de texte. Les Transformers peuvent être optimisés de façon normale avec descente de gradient.

L'architecture d'un Transformer est représenté sur la Fig 1. On peut voir que le modèle est séparé en deux parties qui sont l'encoder (à gauche) et le decoder (à droite). On s'intéressera principalement à l'encoder dans le cadre de ce projet. Plus de détails sont disponibles dans l'article original [1].

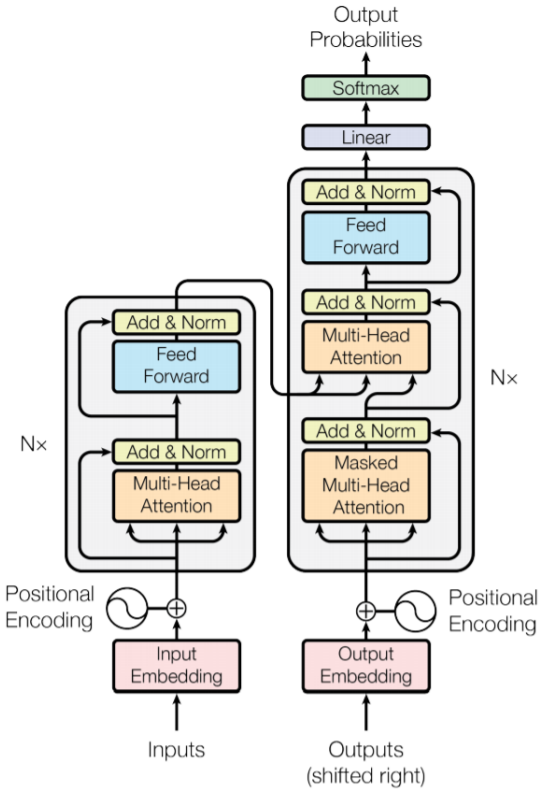

Figure 1. Le Transformer - architecture du modèle [1].

Le Transformer décrit dans l'article [1] est composé de $N = 6$ couches d'encoders, chacune composée de 2

sous-couches: un bloc d'Attention et un simple bloc *Feed Forward*. Le bloc d'Attention permet au modèle de relier tous les items de la séquence d'entrée et de modéliser leur relations. L'architecture contient aussi plusieurs raccourcis (*Residuals*), qui permettent la prise en compte d'informations résiduelles d'avant l'Attention.

La grande particularité des Transformers vient de l'utilisation du mécanisme d'Attention, plus particulièrement de Self-Attention. Ce mécanisme consiste à relier tous les items afin de pouvoir, lors de l'entraînement, optimiser le réseau tout en donnant à chaque items les informations sur tous les autres items de la séquence.

De façon plus formel, soit X la séquence d'entrée de dimension (L, E) avec L la longueur de la séquence (nombre de mots) et E la taille d'embedding de chaque item/mot. La Self-Attention consiste à générer 3 abstractions Q, K, V à partir de X en utilisant les matrices de poids W_Q, W_K, W_V tel que $Q = XW_Q$...etc Une fois ces 3 abstractions générées, on peut calculer l'Attention avec la formule suivante:

$$Z = Attention(Q, K, V) = Softmax(\frac{Q \times K^T}{\sqrt{d_k}})V$$

On obtient ainsi les scores d'Attention pour tous les items de la séquence. En réalité, les Transformers utilisent le mécanisme de *multi-head Attention*, qui consiste à générer $M = 8$ abstractions Q_i, K_i, V_i à l'aide des matrices W_i^Q, W_i^K, W_i^V avec $i \in \{1, ..., M\}$ et à calculer M scores d'Attention Z_i. Les scores Z_i sont ensuite concaténés et on calcule le score d'Attention final $Z = concat(Z_i) \times W_O$ avec W_O la matrice de pondération de la sortie. Z représente donc la pondération de tous les Z_i. La *multi-head Attention* permet au modèle d'avoir une meilleure modélisation et compréhension sémantique de la séquence d'entrée.

L'Attention et les Transformers sont non directionnels, ils ne prennent pas en compte la position des items dans la séquence. Pour corriger cela, une information sur la position est ajoutée à chaque item à l'aide d'une fonction cyclique (*Positional Encoding* in Fig 1). De plus, notons que la complexité de l'Attention est de $O(L^2E)$ [1]. Cela pose rarement des problèmes en pratique, car les calculs d'Attention peuvent être réalisés en parallèle sur GPU, contrairement aux réseaux récurrents dont les opérations sont séquentielles [1, 3, 5].

Le succès des Transformers à donnée naissance à de nombreux modèles, le plus connu étant BERT [2]. Ce modèle composé de 12 encoders superposés est considéré comme étant parmi les meilleurs modèles de traitement de texte en NLP. Cependant, ces modèles sont conçus pour traiter une seule séquence 1D de texte, ce qui ne permet pas d'étudier les relations entre différents titres boursiers. Les Transformers et l'Attention [1] ne peuvent en effet pas modéliser les relations entre les items de plusieurs séquences.

3. Objectifs

Il serait très intéressant d'étudier les relations entre les différents titres et de voir si la prise en compte de ces interactions entre titres permet de mieux prédire l'évolution du marché par rapport aux études se focalisant seulement sur un titre [3, 5, 6].

Pour cela, nous voulons créer un modèle capable de traiter plusieurs titres en même temps. Il s'agirait d'analyser et de mettre en relation leurs informations temporelles et donc de traiter simultanément leurs séquences. Cependant, il n'existe aucune base de données publique pour entraîner un tel modèle puisque aucune base de données publique ne contient les informations standardisées de plusieurs titres, en tout cas a notre connaissance.

Notre projet peut donc être séparé en deux objectifs distincts: la création d'un dataset contenant les informations standardisées de plusieurs titres sur une certaine durée; et la création d'un modèle inspiré des Transformers [1] afin d'analyser et traiter ces différentes séquences simultanément.

3.1. Création du jeu de données

Notre premier objectif consiste à générer un dataset contenant les informations de plusieurs titres sur une certaine durée. Bien que les données financières "récentes" soient facilement accessible en ligne, il est assez compliqué d'obtenir des données historiques fiables gratuitement. La plupart des données historiques en ligne ne viennent souvent que d'un *market maker* particulier qui donne ses données, mais pas les données du marché dans son ensemble. De plus, les données historiques manquent parfois de granularité: on ne peut obtenir que les information mensuelles ou journalières, mais pas plus de détails.

Notre dataset devra contenir des données fiables et standardisées sans valeurs manquantes ou décalages temporels pour pouvoir modéliser les interactions entre les titres.

Le monde de la finance étant très vaste, il faut commencer par choisir les titres que nous allons étudier (combien de titres? lesquels? ont-ils des interactions intéressantes pour nous?) et trouver des sources fiables pour les données brutes de chaque titre. Il faut aussi choisir la période sur laquelle nous allons travailler (1 mois, 5 ans, 15 ans?) ainsi que la fréquence des données (chaque heure/jour/semaine...?). Il faut aussi choisir les valeurs à prendre en compte car le "prix" ne contient pas toute l'information (il y a aussi le volume, le prix d'ouverture, etc...).

Pour générer un dataset utilisable par un modèle de Deep Learning, les informations brutes devront être formatées et standardisées. Il faudra choisir comment représenter ces données multi-dimensionnelles et gérer les données manquantes pour certains instants.

Enfin, le dataset devra facilement être mis à jour et modulable. Nous voulons être capable de rajouter les informations de certains titres, d'en retirer d'autres et de modifier la durée sur laquelle nous récupérons les données. En réalité,

nous n'avons pas pour objectif le dataset en lui-même mais plutôt un script ou programme qui puisse générer un dataset selon les titres et la durée voulus.

3.2. Création du modèle

Une fois le dataset créé, nous avons pour objectif de mettre au point un modèle capable d'utiliser ce dataset. Dans le but de mieux comprendre les relations entre les différents titres boursiers et de prédire leurs évolutions, notre second objectif est de créer un modèle de Deep Learning capable traiter plusieurs séquences boursières simultanément.

On s'inspirera pour cela des récents progrès faits en NLP, notamment avec les Transformers [1] et BERT [2]. Nous pensons qu'il est possible d'utiliser le mécanisme d'Attention pour améliorer les prédictions des modèles financiers. Ce mécanisme peut permettre de mettre en relation tous les items contenus dans une sous partie de notre dataset, ce qui revient à traiter nos différentes séquences en prenant en compte les interactions entre elles. D'un point de vue conceptuel, cela permet de replacer chaque titre dans son contexte financier, de lui donner de l'information sur l'ensemble des titres plutôt que sur lui-même seulement.

Les Transformers ont déjà été appliqués pour aider à prédire le marché, mais seulement pour de l'analyse de sentiments afin de comprendre comment le ressenti sur les réseaux sociaux influence les cours de la bourse [7, 8]. A notre connaissance, aucun article n'a été publié pour se servir des Transformers directement sur les données financières et comprendre les corrélations qu'il y a entre les différentes actions.

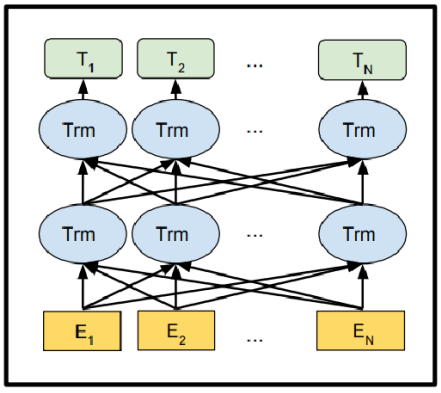

Figure 2. Architecture de BERT [2]

Nous souhaitons nous baser sur l'architecture de BERT [2], qui est montré dans Fig 2. BERT est le state of the art pour la plupart des tâches de traitement de texte et est composé de plusieurs encoders superposés. Vu ses résultats avec une séquence de texte, nous supposons que son architecture sera aussi apte à traiter plusieurs séquences boursières.

Afin de créer le modèle voulu, il sera nécessaire d'adapter le mécanisme d'Attention 1D (celui utilisé dans les Transformers et BERT) afin de pouvoir l'utiliser sur plusieurs séquences simultanément. Cela revient à utiliser un mécanisme d'Attention en 2D [4]. L'attention 2D vise à modéliser les interactions entre tous les items de plusieurs séquences.

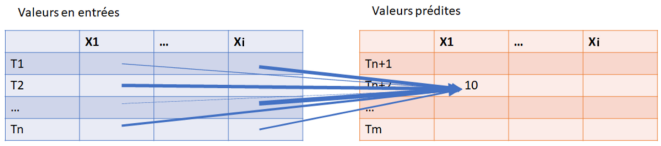

Figure 3. Représentation de l'Attention 2D avec nos données

Fig 3 donne une représentation du fonctionnement de l'Attention 2D [4] voulu où la taille de chaque flèche montre l'influence d'un titre par rapport à un autre, avec X_j les titres de bourses, T_j les différent instants, n le nombre de séquence d'entraînement et $m - n$ le nombre de séquences prédites. Un grosse flèche indique que l'état d'un titre à un certain instant a beaucoup d'influence sur un autre titre à un (autre) instant.

Compte tenu de l'ampleur des tâches à réaliser, nous ne viserons pas forcément un modèle très performant. Nous espérons pourvoir créer un modèle qui puisse utiliser le mécanisme d'Attention 2D et prouver que le concept d'Attention a un fort potentiel dans le domaine financier. Notre modèle agira comme une preuve de concept.

Nos objectifs pour ce projet consistent donc à générer un dataset contenant les informations standardisées de plusieurs séquences boursières, puis à générer un modèle basé sur l'architecture de BERT [2] qui puisse utiliser ce dataset efficacement grâce au mécanisme d'Attention 2D (pour plusieurs séquences).

4. Méthodologie

Cette section décrit les différents aspects de la méthodologie utilisée pour notre projet.

4.1. Jeu de données

Notre premier objectif est donc de créer un dataset contenant l'information standardisée de plusieurs titres sur une durée déterminée. la création du dataset se fera avec les données de Yahoo Finance, car il s'agit du seul fournisseur d'informations financières fiables et gratuites. On s'intéressera aux données financières historiques, afin d'avoir assez d'informations pour éviter un sur apprentissage.

Les données brut de Yahoo Finance seront ensuite traitées et standardisées. Elles seront notamment mise à la même longueur.

En effet, comme nous avons pris pour parti de prendre des données sur une période allant du 30 octobre 2004 au 24 juin 2020 et que certaines compagnies n'existaient pas à ce moment, il faudra retirer certains titres de nos données. Ce choix sera purement subjectif, mais nous essayerons

d'obtenir un bon équilibre entre le nombre de titres à conserver et le nombre de valeurs minimum à avoir.

Après ce premier tri, il faudra aussi gérer des séquences de tailles différentes dû à un manque de données. Afin de pouvoir comparer nos titres, les séquences devront être synchronisées, c'est à dire faire correspondre une dimension à une date. On ne peut pas se permettre d'avoir un décalage temporel entre nos séquences. Comme nous avons voulu garder un maximum de données, nous avons fait de la création de données pour remplacer les valeurs manquantes. Nous avons rajouté les valeurs manquantes par une copie des valeurs précédentes dans la séquence jusqu'à obtenir des séquences de taille 4620 pour tous les titres (la médiane était seulement de 3982).

Toutes les opération seront automatisées pour permettre d'obtenir un dataset modulable et facilement adaptable. On mettra l'accent sur l'utilisation de web-scraping afin de récupérer les données automatiquement et éviter les erreurs humaines.

4.2. Création du modèle

Le modèle sera créé sous Pytorch en suivant le modèle Transformers [1] et en rajoutant des couches pour nous permettre de traiter nos données. Plus précisément, notre modèle suivra fidèlement l'architecture de BERT [2] comme dans Fig 2. Cette architecture a déjà fait ses preuves avec des séquences de textes et pourra peut-être arriver à d'aussi bons résultats avec les séquences financières.

Le modèle est donc constitué d'encoders superposés. Nous avons ainsi codé des encoders nous même en Pytorch. La seule fonction que nous avons utilisée est la fonction MultiHeadAttention de Pytorch. A la base, nous avons une couche d'embedding qui permet d'augmenter notre taille d'embedding via des couches fully-connected. Cette augmentation de la taille de l'embedding est quelque chose de très important car celui des données initiales est assez faible (de taille 11). Or, la taille de l'embedding est ce qui définit la taille du mécanisme d'attention. Plus elle est petite, moins le mécanisme d'attention a de matière pour comprendre les corrélations d'informations. Cette couche est suivie de la partie Encodeur, qui se compose de plusieurs couche encodeur superposées sur elles-mêmes.

Ensuite nous avons deux couches en parallèles : Un classifieur qui permet donner la tendance du titre avec 3 classes possibles: *bearish*, *bullish* et *stable*. Nous avons ensuite la partie régression, qui permet de prédire la valeur d'un titre. L'utilisation de classification et régression simultanément permet une meilleure compréhension sémantique pour les modèles de type BERT [2, 3].

4.3. Entraînement

Nous avons deux loss différentes. La loss pour la classification est une loss Cross-Entropy tandis que celle pour la régression une loss MSE (mean squared error). Nous générons notre propre fonction de loss, qui correspond à une addition de la MSE et Cross-Entropy. Durant l'entraînement, nous utilisons ADAM comme algorithme d'optimisation avec un dropout de 0.1.

Nous utilisons 3 bases différentes, une base train pour entraîner le système, une base validation pour voir si le système performe correctement et qu'il n'y a pas de sur-apprentissage. Une fois que les valeurs de loss obtenues sur la base validation sont correctes, nous passons la base test pour avoir nos résultats. Ces bases sont créées de la même façon: nous sélectionnons une date de départ de façon aléatoire dans le dataset. à partir de cette date, nous prenons les informations des 5 jours qui suivent comment entrée de notre système et nous tentons de prédire les évolutions de chaque titre (en pourcentages) pour les 3 jours suivant.

Pour faciliter l'entraînement, nous utilisons un générateur de batch codé par nos soins, et qui permet d'automatiquement obtenir les informations sur 5 jours consécutifs aléatoires, ainsi que les valeurs attendues pour les 3 prochains jours. Cette fonction de génération de batch permet un entraînement par step plus simple, mais la sélection d'une date de début aléatoire rend inutile la notion d'epoch.

4.4. Matériel

Pour faire tourner notre modèle, nous utilisons comme IDE Jupyter, tous les notebooks sont écrits en Python et nous utilisons la bibliothèque Pytorch qui est une bibliothèque spécialisée dans le Deep Learning. Comme notre modèle est un modèle avec une complexité quadratique en mémoire, nous le faisons tourner sur le serveur JUNA de l'ETS qui possède 2 cartes graphiques Nvidia Tesla P100 avec 16GB de mémoire.

Pour la génération du dataset, nous utilisons le package *Pandas_datareader* afin de faciliter la collecte d'information sur Yahoo Finance.

5. Résultats

Nous allons maintenant présenter les résultats de notre projet.

5.1. Jeu de données

Notre premier résultat est d'avoir généré un dataset contenant les valeurs de 477 titres sur 4620 jours en tout. Ce dataset est standardisé et permet de comparer efficacement les différentes évolutions des titres. Ces données ont été récupéré sur une période de 15 ans, car comme mentionné dans le survey [3], les données de courte période peuvent amener à un sur-apprentissage et ne sont pas vraiment fiables pour faire des prévisions.

Pour simplifier sa représentation, les données sont stockées sous forme de DataFrame 3D de dimensions 4620*477*9 comme montré dans Fig 4. Les 477 titres choisis sont composés de: 430 titres du s&p500, 16 matières premières, 19 monnaies et 12 index (Voir Annexe 2 pour plus de détails). Nous n'avons pris que 430 titres sur les

500 présent dans le s&p500 car certains titres avaient trop de valeurs manquantes, ou des séquences trop courte (c'était par exemple le cas de Facebook, qui est rentré en bourse en 2012, et n'a donc pas assez de données).

	^TNX	^VIX	A	AAP
2004-08-30	[4.2230000495910645, 4.188000202178955, 4.2230...	[15.579999923706055, 15.300000190734865, 15.35...	[15.40772533416748, 15.064377784729006, 15.379...	[25.13333320617676, 24.79999923706055, 24.9133...
2004-08-31	[4.184000015258788, 4.096000194549565, 4.1820...	[15.850000381469727, 15.279999732971193, 15.64...	[15.021459579467773, 14.320457458496096, 15.0,...	[24.979999542236328, 24.6200008392334, 24.9799...
2004-09-01	[4.1599998474121085, 4.081999778747559, 4.0939...	[15.390000343322756, 14.720000267028807, 15.39...	[14.98569393157959, 14.470672607421875, 14.613...	[24.913333892822266, 24.63999938964844, 24.706...
2004-09-02	[4.196000099182129, 4.116000175476073, 4.13199...	[15.050000190734865, 14.18000030517578, 14.970...	[15.236051559448242, 14.570815086364744, 14.80...	[24.893333435058594, 24.646667480468875, 24.893...
2004-09-03	[4.297999858856201, 4.165999889373778, 4.18200...	[14.380000114440918, 13.789999961853027, 14.35...	[15.135908126831055, 14.484978675842285, 15.12...	[25.02666664123535, 24.659999847412111, 24.7399...
2004-09-05	[4.297999858856201, 4.165999889373778, 4.18200...	[14.380000114440918, 13.789999961853027, 14.35...	[15.135908126831055, 14.484978675842285, 15.12...	[25.02666664123535, 24.659999847412111, 24.7399...
2004-09-06	[4.297999858856201, 4.165999889373778, 4.18200...	[14.380000114440918, 13.789999961853027, 14.35...	[15.135908126831055, 14.484978675842285, 15.12...	[25.02666664123535, 24.659999847412111, 24.7399...
2004-	[4.293000221252441,	[14.640000343322756,	[14.97138786315918,	[25.29333305358887,

Figure 4. Échantillon tronqué de notre dataset

Les données sont récupérées par web-scraping sur Yahoo Finance, qui est considéré comme la source gratuite la plus fiable pour des données financières. Nous avons créé un script Python qui récupère automatiquement les informations de plusieurs titres sur une durée voulue. Notre dataset est composé d'informations journalière car c'est la fréquence d'actualisation la plus courte disponible gratuitement sur Yahoo Finance. Les données sont fiables et aucune information n'a été retiré afin de ne pas priver le modèle d'informations potentiellement utiles. Une partie non négligeable de l'information est "artificielle" afin de permettre la comparaison entre les données (voir Annexe 1).

Chaque information journalière contient 11 valeurs. Parmi ces 11 valeurs, 7 sont des informations financières (prix d'ouverture, de fermeture, le prix minimum, prix maximum, volume, fermeture ajustée, changement journalier). Les 4 autres valeurs ont été rajouté pour permettre au modèle d'avoir des informations supplémentaires. Notre script rajoute ainsi une valeur pour marquer si les données sont réelles ou générées pour remplir une valeur manquante (voir Annexe 1).

Vu que les modèles à base d'Attention sont non-directionnels, les 3 valeurs restantes sont utilisées pour représenter l'appartenance des valeurs à un certain titre et des les positionner dans le temps. On rajoute donc des informations pour marquer la position en 2D dans notre dataset.

La récupération des données, l'élimination des données possédant trop de valeurs manquantes, la génération de données artificielles pour combler les valeurs manquantes, la mise en forme et standardisation ainsi que l'ajout des informations supplémentaires sont tous fait automatiquement. Notre script Python permet de générer un dataset financier pour les titres et la période voulus. Notre dataset est donc facile à mettre à jour et maintenir, mais aussi à modifier (ajout de nouveaux titres...). Il peut automatiquement prétraiter les données et s'assurer que le dataset final est standardisé. Le réel résultat ici est notre script Python qui permet d'effectuer toutes ces tâches automatiquement et d'éviter les erreurs humaines. Nous pouvons ainsi obtenir les informations de Yahoo Finance pré-traitées et standardisés en quelques secondes.

5.2. Modèle

Nous avons réussi à créer un modèle de type BERT composé de plusieurs encoders superposés, et pouvant à la fois faire de la régression et de la classification. Le modèle en lui même est fidèle a l'architecture de BERT et utilise le mécanisme d'attention mais il est adapté pour traiter nos données.

L'architecture du modèle final est disponible dans Fig 5. Notre modèle final est composé d'une couche d'embedding qui augmente la dimensionnalité de nos entrées, puis de 3 encoders superposés dont l'architecture est fidèle à celle des Transformers [1], et enfin de nos couches de sorties qui sont une couche de classification et une de régression.

Les couches d'embedding consistent en 5 couches linéaires avec GELU comme fonction d'activation. Les couches de classification consistent en 3 couches Linéaires avec RELU comme activation et celles de régression en 3 couches linéaires avec SoftSign comme fonction d'activation (pour garder les valeurs entre -1 et 1).

Nous n'avons utilisé que 3 encoders au lieu des 12 de BERT [2], car nous n'avions pas les ressources pour supporter un modèle aussi lourd. De plus, notre but est avant tout de montrer un concept, pas d'obtenir les meilleurs performances à tout prix.

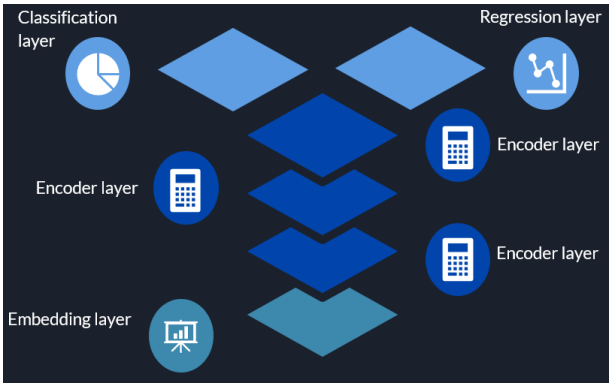

Figure 5. Schéma de l'architecture de notre modèle

Le modèle a été entraîné sur notre dataset pendant plus de 5000 steps avec une taille d'embedding de 128, 8 heads d'Attention par encoder et les informations de chaque titre sur 5 jours. La taille d'embedding de 128 a été choisi car c'était la taille maximum que l'on pouvait supporter sans faire crasher le serveur à cause de la demande en RAM.

Malheureusement et comme le montre Fig 6, les résultats sont plutôt mitigés. D'un coté, le modèle converge, ce qui montre que notre architecture fonctionne et qu'il n'y a , à priori, pas d'erreurs de programmation ni de bug dans notre pipeline. Cependant, le modèle converge très vite et on atteint rapidement un plateau où les performances stagnent.

La partie régression du modèle semble bien fonctionner vu que la fonction de loss MSE est très basse, mais la

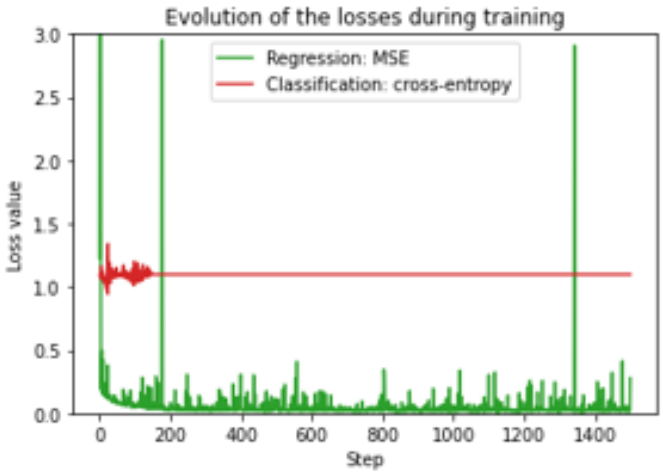

Figure 6. Évolution des fonctions de loss jusqu'à 1500 steps d'entraînement

fonction loss pour la classification stagne trop rapidement et ne descend pas assez.

Il faut savoir que notre modèle était assez léger à ce moment là puisqu'il n'avait que 3 encoders, ce qui lui donnait environ 2 millions de paramètres. Nous avons donc voulu augmenter la complexité du modèle pour voir si l'on pouvait obtenir de meilleurs performances, par exemple en rajoutant des encoders supplémentaires ou en augmentant la taille d'embedding.

Malheureusement nous n'avons pas pu tester nos modèle plus complexes car nous n'avions simplement pas assez de ressource et notamment de RAM. Même si les modèles n'avaient pas beaucoup de paramètres, l'utilisation de l'Attention rend leur complexité quadratique en terme de mémoire. Plus précisément, nous n'avons pas réussi à dépasser une taille d'embedding de 128 et n'avons pas réduit la durée de collecte d'information en dessous de 3 jours pour éviter un simple copié collé des informations de la veille.

Le simple fait de rajouter un encoder supplémentaire ou d'augmenter la taille d'embedding faisait que notre modèle nécessitait plusieurs GB de RAM supplémentaires. Les résultats présentés ici sont ceux obtenus avec notre modèle le plus complexe (embedding = 128, 3 encoders, 8 heads d'Attention), qui demandait déjà environ 15 GB de RAM pour fonctionner.

Encore une fois, le modèle que l'on présente ici est donc plus une sorte de proof of concept, pour montrer que les Transformers [1] peuvent être appliqués sur des séquences financières, mais nous n'avons pas pu poussé notre modèle jusqu'au bout a cause du manque de RAM.

6. Discussion

Comme expliqué plus haut, notre modèle a des résultats qui sont très mitigés. Les fonctions de loss convergent, notamment la loss sur la régression mais la loss de classification a du mal à converger convenablement.

Nos résultats montrent ainsi que l'Attention peut bel et bien être appliquée à plusieurs séquences boursières. Pour l'instant nous ne faisons qu'un proof of concept et nous avons montré que notre technique est fonctionnelle même si nous avons besoin d'avoir de meilleurs résultats pour qu'elle puisse être utilisée en industrie.

Nous pensons que ces résultats sont dû au manque de complexité de notre modèle. Notre modèle ne contenait en effet que 2 millions de paramètres (avec 3 encodeurs et un embedding de 128), tandis que BERT en contient 120 [2] par exemple. Ce manque de complexité en terme de paramètres réduit la capacité d'analyse et de compréhension de notre modèle.

En termes de mémoire utilisée, chaque couche d'Attention a une complexité quadratique selon la longueur de la séquence fournie à l'Attention. Or, la nombre de valeurs à faire passer dans le mécanisme d'Attention correspond à $taille_batch = num_jour \times 477$ car nous avons 477 titres. Cela signifie que chaque jour d'informations supplémentaire et chaque titre supplémentaire dans les batchs d'entraînement ont un impact quadratique sur la quantité de mémoire utilisée.

Nous devons nous contenter de prendre des batchs de données sur 5 jours pour prédire les 3 suivants. Cette limitation de la mémoire nous donne des résultats qui sont plus que mitigés. Cependant la loss converge jusqu'à un certains points, et nous avons remarqué que moins les données d'entrées comprenaient de jours, moins la loss convergeait.

Nous devons donc trouver un bon équilibre pour la quantité de jours à utiliser, entre performance du modèle et consommation de mémoire. Utiliser plus de jours en entrée donnera sûrement de meilleurs résultats, mais augmentera grandement la consommation de mémoire. Conceptuellement, utiliser moins de jours risque au contraire de réduire la capacité d'apprentissage sur l'évolution des cours à long et moyen terme. Le modèle risque alors de se concentrer exclusivement sur les résultats de la veille, qui peuvent ne pas être suffisant pour comprendre la non-linéarité du marché.

Nous avons également un problème sur la taille d'embedding qui est trop faible, 128, alors que celle utilisée pour BERT est de 512 [2]. Il est probable que l'augmentation de l'embedding permette d'augmenter les performances du modèle.

Afin d'améliorer notre modèle, nous devons donc réussir à réduire la quantité de mémoire utilisée tout en augmentant le nombre de jours et la taille d'embedding. La technique la plus simple serait de retirer entièrement certains titres.

Cependant, la suppression de titres peut être mauvais pour la compréhension de la bourse et également nous aurons un problème à choisir quel titre supprimer. Nous pouvons également changer complètement notre idée et ne se contenter de prédire qu'un seul titre à partir des informations d'autres titres, ce qui réduira sûrement le nombre de titre nécessaire à prendre en compte.

Bien que retirer des titres soit attractif, il faudrait au préalable étudier lesquels conserver et sur lesquels se baser,

ce qui demande une compréhension poussée des marchés et surtout des titres en question. N'ayant pas les compétences pour faire ce genre de choix, nous préférons conserver tous les titres pour ce projet et juste présenter notre "proof of concept".

Il y a également un autre problème qui est intrinsèque à la bourse. La bourse s'adapte constamment aux acteurs en jeu, à la géopolitique et aux nouvelles techniques. Les investisseurs, spéculateurs veulent toujours être devant les autres et changent leur façon de "jouer" constamment.

Il est possible qu'il n'y a pas de pattern à long terme et qu'il soit plus intéressant de se concentrer sur de la prédiction à très court termes, avec une fréquence de données bien plus élevé qu'une fréquence quotidienne. La bourse s'adapte, et les schémas/modèles d'il y a 10 ans ne sont plus forcément valables pour modéliser la bourse aujourd'hui. La bourse est un système complexe avec beaucoup d'acteurs en compétition, qui s'adaptent en continu aux autres. Par définition, un tel système complexe pourrait donc ne pas avoir de pattern sur le long terme, ce qui correspond pourtant exactement à ce que nous cherchons.

7. Conclusion

Dans ce projet, nous avons voulu créer un nouveau modèle pour faire de la prédiction boursière. A la différence des autres modèles de prédiction qui prennent en compte un ou quelques titres boursiers pour faire la prédiction, nous en avons créé un capable de gérer plusieurs centaines de titres simultanément. Nous supposons que la bourse est un environnement complexe et que les titres de bourses ont des corrélations complexes entres eux.

Notre modèle a été créé en se basant sur l'état de l'art du traitement de texte en NLP, les modèles Transformers et BERT. Ces modèles sont constitués de plusieurs couches d'Attention qui permettent de corréler l'informations des données entres elles. Notre modèle permet deux choses, il fait de la prédiction via une couche de régression et il fait également de la classification, pour savoir si le titre va monter ou descendre via une couche de classification.

Afin d'entraîner notre modèle, nous générons un dataset standardisé qui contient les informations temporelles de plusieurs titres boursiers. Le traitement des données est automatisé, avec récupération des données par web-scraping et pré-traitement par un script que nous avons codé.

Nous montrons ainsi qu'il est possible d'appliquer le mécanisme d'Attention à plusieurs séquences boursières. Notre modèle est une proof of concept, il n'a jamais été fait avant et notre projet montre qu'il possède du potentiel, même si, malheureusement, les résultats sont mitigés. Nous attribuons ces résultats au manque de ressource dû à la complexité du modèle en termes de mémoire. Les couches d'Attentions ont une complexité quadratique en mémoire ce qui nous force à limiter la taille des séquences en entrées.

Afin d'améliorer les performances de notre modèle, nous pensons qu'il faudrait limiter le nombre de jours de données d'entraînement et redéfinir les données sur lesquels nous travaillons, ainsi qu'augmenter la complexité (en terme de paramètres) de notre modèle, notamment via un embedding plus grand. Cela nécessite cependant une bonne gestion des ressource: la suite du projet consiste à mieux gérer la consommation de mémoire de notre modèle pour améliorer ses performances.

Acknowledgments

Nous souhaitons remercier notre professeure Sylvie Ratté pour nous avoir permis de travailler sur ce projet, ainsi que Christopher Brunet, doctorant en économie, qui nous a aidé à mieux comprendre les rudiments des modèles financiers.

References

[1] Ashish Vaswani and al, *Attention Is All You Need*, presented at the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA. [Online] Available: https://arxiv.org/pdf/1706.03762.pdf. [Accessed: 29-Jul-2020]

[2] Jacob Devlin and al, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2018. [Online] Available: https://arxiv.org/pdf/1810.04805.pdf. [Accessed: 29-Jul-2020]

[3] Weiwei Jiang, *Applications of deep learning in stock market prediction: recent progress*, 29 février 2020. [Online] Available: https://arxiv.org/pdf/2003.01859.pdf. [Accessed: 29-Jul-2020]

[4] Niki Parmar and al *Image Transformer*, 15 juin 2018. [Online] Available : https://arxiv.org/pdf/1802.05751.pdf. [Accessed: 29-Jul-2020]

[5] D. M. Q. Nelson, A. C. M. Pereira and R. A. de Oliveira, *Stock market's price movement prediction with LSTM neural networks*, 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 1419-1426, doi: 10.1109/IJCNN.2017.7966019.

[6] Jia Wang et al. *CLVSA: A Convolutional LSTM Based Variational Sequence-to-Sequence Model with Attention for Predicting Trends of Financial Markets*, 2019. [Online] Available : https://www.ijcai.org/Proceedings/2019/0514.pdf. [Accessed: 29-Jul-2020]

[7] Guang Liu, Xiaojie Wang, *A Numerical-Based Attention Method for Stock Market Prediction With Dual Information*, 12 décembre 2018.

[8] Jintao Liu et al. *Transformer-Based Capsule Network For Stock Movements Prediction*, 12 aout 2019. [Online] Available: https://www.aclweb.org/anthology/W19-5511.pdf. [Accessed: 29-Jul-2020]

Annexe 1

Dans cette annexe, nous allons montrer comment nous ”fusionnons” les informations de plusieurs titres en un dataset. Afin de créer un dataset permettant de comparer plusieurs séquences, on concatène les séquences selon leurs dates comme montré dans Fig 7.

Date	Value 1	...	Value m
T1	A11	...	A1m
T2	A21	...	A2m
T5	A51	...	A5m
T6	A61	...	A6m
T9	A91	...	A9m
...	...	...	...
Tn	An1	...	Anm

Stock A

Date	Value1	...	Value m
T1	B11	...	B1m
T2	B21	...	B2m
T3	B31	...	B3m
T4	B41	...	B4m
T8	B81	...	B8m
...	...	...	...
Tn	Bn1	...	Bnm

Stock B

Date	Stock A	Stock B
T1	[A11,...,A1m, 1]	[B11,...,B1m, 1]
T2	[A21,...,A2m, 1]	[B21,...,B2m, 1]
T3	Nan	[B31,...,B3m, 1]
T4	Nan	[B41,...,B4m, 1]
T5	[A51,...,A5m, 1]	Nan
T6	[A61,...,A6m, 1]	Nan
...	...	...

Figure 7. Processus de concaténation avec remplissage de valeurs manquantes

Certains titres n’ont pas les mêmes fréquences d’actualisation, on se retrouve donc avec des Nan. On duplique la valeur précédente pour retirer les Nan, et on ajoute une valeur pour indiquer si les valeurs sont copiées ou originelles. Ce processus est montré dans Fig 8.

Date	Stock A	Stock B
T1	[A11,...,A1m, 1]	[B11,...,B1m, 1]
T2	[A21,...,A2m, 1]	[B21,...,B2m, 1]
T3	[A21,...,A2m, 0]	[B31,...,B3m, 1]
T4	[A21,...,A2m, 0]	[B41,...,B4m, 1]
T5	[A51,...,A5m, 1]	[B41,...,B4m, 0]
T6	[A61,...,A6m, 1]	[B41,...,B4m, 0]
...	...	...

Figure 8. Doublement des valeurs afin de remplacer les Nan

Annexe 2

Nous donnons ici une liste exhaustive de tous les titres inclus dans notre dataset. Ces titres ont tous servis à entraîner notre modèle et à l'évaluer.

Ci dessous la liste des 477 titres dans notre dataset: 430 titres du S&P500, 16 matières premières, 19 monnaies, 12 index: ['BO=F', 'C=F', 'FC=F', 'KW=F', 'LB=F', 'LC=F', 'LH=F', 'O=F', 'S=F', 'SB=F', 'CL=F', 'HO=F', 'NG=F', 'GC=F', 'HG=F', 'SI=F', 'BHD=X', 'CAD=X', 'CHF=X', 'CNY=X', 'EUR=X', 'GBP=X', 'HKD=X', 'INR=X', 'JOD=X', 'JPY=X', 'KRW=X', 'KWD=X', 'MXN=X', 'NOK=X', 'NZD=X', 'RUB=X', 'SEK=X', 'SGD=X', 'ZAR=X', 'ÎBVSP', 'ÎDJI', 'ÎFCHI', 'ÎGDAXI', 'ÎGSPC', 'ÎHSCE', 'ÎHSI', 'ÎIXIC', 'Î225', 'ÎRUT', 'ÎTNX', 'ÎVIX', 'A', 'AAP', 'AAPL', 'ABC', 'ABMD', 'ABT', 'ACN', 'ADBE', 'ADI', 'ADM', 'ADP', 'ADS', 'ADSK', 'AEE', 'AEP', 'AES', 'AFL', 'AIG', 'AIV', 'AIZ', 'AJG', 'AKAM', 'ALB', 'ALGN', 'ALK', 'ALL', 'ALXN', 'AMAT', 'AMD', 'AME', 'AMG', 'AMGN', 'AMT', 'AMZN', 'ANSS', 'ANTM', 'AON', 'AOS', 'APA', 'APD', 'APH', 'ARE', 'ATO', 'ATVI', 'AVB', 'AVY', 'AXP', 'AYI', 'AZO', 'BA', 'BAC', 'BAX', 'BBY', 'BDX', 'BEN', 'BF-B', 'BIIB', 'BK', 'BKNG', 'BKR', 'BLK', 'BLL', 'BMY', 'BRK-B', 'BSX', 'BWA', 'BXP', 'C', 'CAG', 'CAH', 'CAT', 'CB', 'CBRE', 'CCI', 'CCL', 'CDNS', 'CERN', 'CHD', 'CHK', 'CHRW', 'CI', 'CINF', 'CL', 'CLX', 'CMA', 'CMCSA', 'CME', 'CMI', 'CMS', 'CNC', 'CNP', 'COF', 'COG', 'COO', 'COP', 'COST', 'CPB', 'CPRT', 'CRM', 'CSCO', 'CSX', 'CTAS', 'CTL', 'CTSH', 'CTXS', 'CVS', 'CVX', 'D', 'DD', 'DE', 'DGX', 'DHI', 'DHR', 'DIS', 'DISH', 'DLTR', 'DOV', 'DPZ', 'DRE', 'DRI', 'DTE', 'DUK', 'DVA', 'DVN', 'DXC', 'EA', 'EBAY', 'ECL', 'ED', 'EFX', 'EIX', 'EL', 'EMN', 'EMR', 'EOG', 'EQIX', 'EQR', 'EQT', 'ES', 'ESS', 'ETFC', 'ETN', 'ETR', 'EVRG', 'EW', 'EXC', 'EXPD', 'EXR', 'F', 'FAST', 'FCX', 'FDX', 'FE', 'FFIV', 'FIS', 'FISV', 'FITB', 'FL', 'FLIR', 'FLR', 'FLS', 'FMC', 'FRT', 'FTI', 'GD', 'GE', 'GILD', 'GIS', 'GL', 'GLW', 'GOOG', 'GOOGL', 'GPC', 'GPN', 'GPS', 'GRMN', 'GS', 'GT', 'GWW', 'HAL', 'HAS', 'HBAN', 'HD', 'HES', 'HFC', 'HIG', 'HOG', 'HOLX', 'HON', 'HP', 'HPQ', 'HRB', 'HRL', 'HSIC', 'HST', 'HSY', 'HUM', 'IBM', 'IDXX', 'IEX', 'IFF', 'ILMN', 'INCY', 'INTC', 'INTU', 'IP', 'IPG', 'IRM', 'ISRG', 'IT', 'ITW', 'IVZ', 'J', 'JBHT', 'JCI', 'JKHY', 'JNJ', 'JNPR', 'JPM', 'JWN', 'K', 'KEY', 'KIM', 'KLAC', 'KMB', 'KMX', 'KO', 'KR', 'KSS', 'KSU', 'L', 'LB', 'LEG', 'LEN', 'LH', 'LHX', 'LIN', 'LKQ', 'LLY', 'LMT', 'LNC', 'LNT', 'LOW', 'LRCX', 'LUV', 'M', 'MAA', 'MAC', 'MAR', 'MAS', 'MAT', 'MCD', 'MCHP', 'MCK', 'MCO', 'MDLZ', 'MDT', 'MET', 'MGM', 'MHK', 'MKC', 'MLM', 'MMC', 'MMM', 'MNST', 'MO', 'MOS', 'MRK', 'MRO', 'MS', 'MSFT', 'MSI', 'MTB', 'MTD', 'MU', 'MXIM', 'MYL', 'NBL', 'NDAQ', 'NEE', 'NEM', 'NFLX', 'NI', 'NKE', 'NOC', 'NOV', 'NRG', 'NSC', 'NTAP', 'NTRS', 'NUE', 'NVDA', 'NVR', 'NWL', 'O', 'ODFL', 'OKE', 'OMC', 'ORCL', 'ORLY', 'OXY', 'PAYX', 'PBCT', 'PCAR', 'PCG', 'PDCO', 'PEAK', 'PEG', 'PEP', 'PFE', 'PFG', 'PG', 'PGR', 'PH', 'PHM', 'PKG', 'PKI', 'PLD', 'PNC', 'PNR', 'PNW', 'PPG', 'PPL', 'PRGO', 'PRU', 'PSA', 'PVH', 'PWR', 'PXD', 'QCOM', 'RCL', 'RE', 'REG', 'REGN', 'RF', 'RHI', 'RJF', 'RL', 'RMD', 'ROK', 'ROL', 'ROP', 'ROST', 'RRC', 'RSG', 'RTX', 'SBAC', 'SBUX', 'SCHW', 'SEE', 'SHW', 'SIG', 'SIVB', 'SJM', 'SLB', 'SLG', 'SNA', 'SNPS', 'SO', 'SPG', 'SPGI', 'SRCL', 'SRE', 'STE', 'STT', 'STX', 'STZ', 'SWK', 'SWKS', 'SYK', 'SYY', 'T', 'TAP', 'TFC', 'TFX', 'TGT', 'TIF', 'TJX', 'TMO', 'TPR', 'TROW', 'TRV', 'TSCO', 'TSN', 'TTWO', 'TXN', 'TXT', 'UDR', 'UHS', 'UNH', 'UNM', 'UNP', 'UPS', 'URI', 'USB', 'VAR', 'VFC', 'VLO', 'VMC', 'VNO', 'VRSN', 'VRTX', 'VTR', 'VZ', 'WAB', 'WAT', 'WBA', 'WDC', 'WEC', 'WELL', 'WFC', 'WHR', 'WLTW', 'WM', 'WMB', 'WMT', 'WRB', 'WST', 'WY', 'WYNN', 'XEC', 'XEL', 'XLNX', 'XOM', 'XRAY', 'XRX', 'YUM', 'ZBH', 'ZBRA', 'ZION']