5A ModIA

Rapport de Projet

Prédiction de l'énergie moléculaire

Elèves :
Karima GHAMNIA
Cassandra MUSSARD

Enseignant :
Sixin ZHANG

12 janvier 2025

Table des matières

1 Introduction

L'objectif de ce projet est de prédire l'énergie moléculaire en se basant sur la structure géométrique des molécules ainsi que sur des données supplémentaires relatives aux atomes, telles que leur nombre atomique.

Le projet représente un problème de grande dimension, que nous cherchons à résoudre à l'aide de méthodes avancées de machine learning, en tenant compte des contraintes liées à la représentation en 3D.

2 Data Processing

2.1 Dataset

Le jeu de données utilisé est un sous-ensemble du jeu de données **QM7-X**(extension de QM7), comprenant 4739 structures de molécules avec différents nombres d'atomes. Les données sont organisées en deux dossiers principaux : "atoms" pour les configurations moléculaires et "energies" pour les valeurs d'énergie du train stockées dans un fichier `.csv`. Notre but est de produire un fichier similaire pour les données de test, grâce aux prédictions obtenues par un modèle de machine learning.

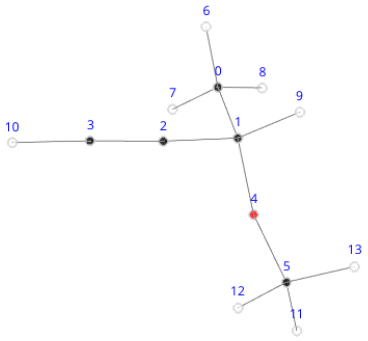


FIGURE 1 – Visualisation d'une des molécules du dataset

2.2 Analyse de données

3 Méthodes

D'abord, nous avons implémenté un réseau MLP simple en utilisant les matrices de Coulomb des molécules comme entrée, plutôt que les positions directes des atomes.

Cette étape reproduit les travaux de [2].

Ensuite, nous avons expérimenté avec la méthode de scattering 3D combinée à différentes régressions linéaires [3].

Enfin, nous avons essayé d'implémenter diverses autres méthodes issues de la littérature dans l'espoir d'améliorer notre score public sur Kaggle.

3.1 Méthode 1 : Matrice de Coulomb

La matrice de Coulomb est une représentation matricielle d'une molécule utilisée en chimie pour capturer des informations sur sa structure électronique et géométrique. Elle est construite à partir des charges nucléaires Z_i et des coordonnées cartésiennes R_i des atomes , comme suit :

$$C_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \text{si } i = j, \\ \frac{Z_i Z_j}{|R_i - R_j|} & \text{si } i \neq j. \end{cases}$$

Les éléments de la diagonale représentent l'énergie d'un atome lorsqu'il est isolé c'est à dire lorsqu'il n'est pas en interaction avec les autres atomes de la molécule.
Les éléments en dehors de la diagonale quant à eux, représentent la répulsion de Coulomb entre toutes les paires possibles de noyaux dans la molécule.

L'objectif de cette méthode est de faire une classification de l'énergie pour chaque molécule à l'aide de cette matrice de Coulomb.

3.1.1 Création du dataset

Dans un premier temps nous avons crée notre dataset d'entraînement et de test en définissant une classe Pytorch.

Pour chaque molécule nous récupérons les atomes ainsi que leurs positions (x, y, z). À partir de ces deux informations, nous reconstruisons la molécule à l'aide du package rdkit.
Ensuite, nous avons calculé nos matrices de Coulomb avec le package `"deepchem"`.

Le premier problème rencontré à été d'avoir des **molécules avec un nombre d'atomes différent**. En effet, cela implique d'avoir une matrice de Coulomb de taille différente.
Pour pallier à ce problème nous avons calculer le nombre d'atomes maximal dans toutes les molécules du dataset. Nous avons ensuite appliqué du zéro padding sur les molécules

dont le nombre d'atomes est inférieur au nombre d'atomes maximal. Ceci nous permet d'avoir des matrices de Coulomb de taille 23x23.

Enfin, nous séparons notre dataset d'entraînement en 0.8 pour les données d'entraînement et 0.2 pour le dataset de validation. Ceci nous permet de contrôler l'entraînement de notre modèle.

3.1.2 Augmentation de données

Comme expliqué dans [2], nous appliquons la formule suivante sur la 3ème dimension de notre matrice en prenant $\theta = 1$:

$$x = \left[\ldots, \tanh\left(\frac{C-\theta}{\theta}\right), \tanh\left(\frac{C}{\theta}\right), \tanh\left(\frac{C+\theta}{\theta}\right), \ldots\right]$$

Cette méthode est une façon de réaliser de l'augmentation de données. En effet, la fonction tanh est une fonction non linéaire qui comprime ses entrées dans l'intervalle $[-1, 1]$. La formule crée une série de points en variant C autour de C, C-θ, C+θ. Cela permet de simuler des perturbations dans les données d'entrée. Ces perturbations peuvent imiter des variations naturelles dans les données, augmentant ainsi la diversité des exemples d'entraînement sans avoir besoin de collecter des données supplémentaires.

Dans notre code nous proposons aussi une autre méthode d'augmentation de données pour à la fois augmenter le nombre d'échantillons d'entraînement et pour que notre modèle soit plus robuste. Nous avons ajouté du bruit sur les positions des atomes de la molécule.
L'ajout de bruit aux positions des atomes peut aider à rendre le modèle plus robuste. En apprenant à prédire l'énergie d'une molécule à partir de positions légèrement perturbées, le modèle devient moins sensible aux petites variations dans les données d'entrée. De plus, cela aide le modèle à généraliser mieux, car il a été exposé à une variété plus large de positions atomiques. Ceci peut être intéressant si le modèle envisagé est un Multi-Layer Perceptron car il nous faut un nombre conséquant de données pour l'entraînement.

3.1.3 Invariance

Le deuxième problème que nous avons rencontré est un problème lié à l'inversion des lignes de la matrice de Coulomb. En effet, si nous inversons les lignes de cette matrice nous pouvons obtenir une toute autre molécule qui n'aura pas la même énergie que celle

considérée au début. Il faut donc que cette matrice soit invariante aux permutations. Pour rendre la matrice invariante nous utilisons la technique de Sorted Coulomb Matrix mentionnée dans [2]. Cette méthode consiste à ordonner les atomes de la manière suivante :

$$\forall i,\ \|C_i\| \geq \|C_{i+1}\|$$

3.1.4 Modèle

En observant les résultats de [2] nous nous sommes aperçues que les meilleurs résultats étaient obtenus en utilisant un Multi-Layer Perceptron (MLP). Nous avons donc décidé d'implémenter cette méthode.

Nous avons utilisé 4 couches Linéaires chacune suivie d'une leakyReLU (sauf la dernière) afin de garder les valeurs négatives sur les gradients.
Nous avons aussi initialisé les poids du réseau avec une distribution normale de moyenne 0 et d'écart type $\frac{1}{\sqrt{N}}$. Ceci permet de maintenir les activations et les gradients dans des plages raisonnables au début de l'entraînement.

3.1.5 Entraînement

Nous avons entraîné notre modèle sur 12 epochs avec l'optimiseur Adam et un learning rate de 0.01.
Nous avons utilisé la RMSE comme loss et pour éviter toute forme d'overfitting nous avons ajouté un terme de régularisation L2 dans l'optimiseur.
Nous avons alors obtenu ce graphique qui représente la loss au cours des itérations pour le dataset d'entraînement et de validation.

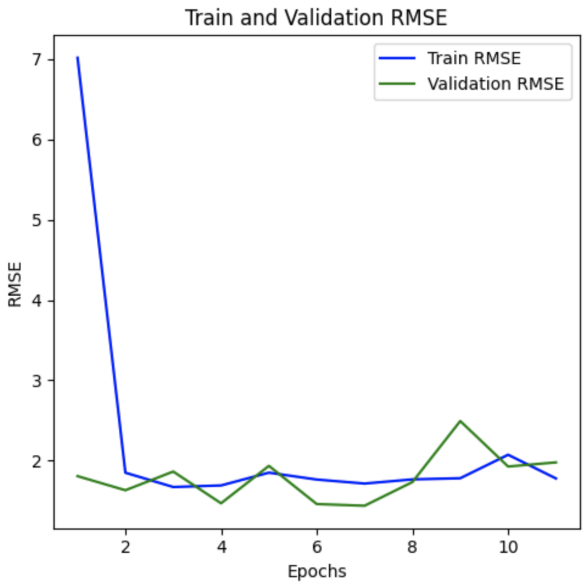


FIGURE 2 – Évolution de la loss d'entraînement et de validation au cours des epochs

Nous remarquons que la loss RMSE varie énormément entre 2 et 3 pour le dataset de validation montrant que l'entraînement n'est pas très stable et pouvant indiquer un comportement d'overfitting car la loss d'entraînement continue de descendre. Il semblerait qu'il y faut arrêter l'entraînement à l'epoch 8 car la loss d'entraînement commence à augmenter.

3.1.6 Résultats

Nous avons dans un dernier temps évaluer notre modèle sur le dataset de test dont nous n'avons pas les labels en utilisant la méthode décrite précédemment.
Sur Kaggle nous avons obtenu un score de 1.56. Ce score est moins élevé que celui obtenu avec le dataset de valdiation mais montre encore une valeur assez élevé et suggère l'utilisation d'une autre méthode pour traiter ce problème.

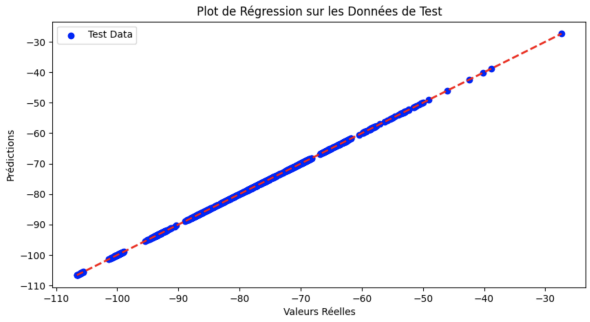


FIGURE 3 – Régression sur les données de test

Nous voyons sur ce graphique que nous arrivons à avoir une relation totalement linéaire et nous remarquons que nous n'avons pas d'outliers ce qui indique une assez bonne performance de notre modèle.

Nous allons maitenant décrire la méthode de 3DScattering qui nous a permis d'avoir de meilleurs résultats que ceux obtenus avec la matrice de Coulomb.

4 Méthode 2 : 3D Scattering

Dans cette partie, nous testons une nouvelle représentation invariante aux rotations et translations tridimensionnelles des molécules. Cette représentation est multi-échelle, basée sur des ondelettes harmoniques solides, et consiste à calculer les coefficients de scattering.

Les coefficients de scattering décomposent les molécules en différentes échelles et fréquences, capturant ainsi leurs caractéristiques globales et locales. Ces coefficients ont des ordres différents présentant leur niveau de précision comme suit :

— **Ordre 0 :** C'est l'ordre le moins précis, car c'est une moyenne statistique. Il est défini par :

$$\int_{\mathbb{R}^3} \rho_x^q(u)\, dx$$

avec ρ_x qui représente les coefficients obtenus après l'application de l'ondelette.

— **Ordre 1 :** Cet ordre ($S_{rho}[j, l, q]$) fournit des descriptions invariantes des états qui sont stables face aux déformations des positions atomiques. De plus, il sépare les échelles du système en agrégeant des motifs géométriques liés à l'agencement des atomes et des liaisons à chaque échelle. La puissance $q = 1$ varie linéairement avec le nombre de particules électroniques représentées par la densité ρ, tandis que $q = 2$ encode les interactions par paires, qui sont liées aux interactions électrostatiques de Coulomb présentée précédemment.

— **Ordre 2 :** Les coefficients de cet ordre ($S_{rho}[j, j', l, q]$)sont invariants aux mouvements rigides de ρ, stables face aux déformations de l'état, et couplent deux échelles au sein du système atomistique, donnant ainsi plus de précision.

4.1 Création du dataset et analyse de données

4.1.1 Création du dataset

La création des dataset "train" et "test" est réalisée en regroupant les différents ordres (0, 1, et 2) des coefficients de scattering.

Les calculs de ces coefficients se font en 2 étapes principales. Premièrement, nous cherchons une représentation spatiale des distributions de charges en calculant une carte de densité des différentes molécules à partir de notre dataset. Ensuite, nous appliquons une transformation de scattering harmonique 3D à cette carte de densité qui consiste à appliquer une ondelette, puis un opérateur non linéaire qui permet de garder l'information au-delà de l'ordre 0 (ordre1, et 2), car comme vu en cours, avec une simple ondelette nous perdons des informations importantes (moyenne nulle).

Concernant la transformation d'ondelette utilisée, nous avons utilisé l'ondelette solid harmonique. Ses propriétés mathématiques permettent d'avoir l'invariance par rotation et translation.

Ces étapes ont été effectuées grâce au tutorial [1].

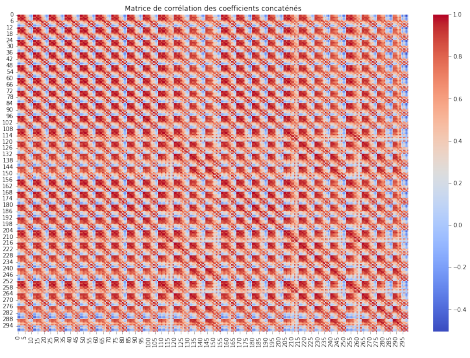
4.1.2 Analyse de données

Dans cette partie nous avons réalisé une analyse des données sur les coefficients de scattering du dataset d'entraînement dans le but de voir si il est nécessaire de réaliser des étapes de pré-processing des données.
La figure 4a représente la matrice de corrélation entre les coefficients de scattering. Nous observons qu'il y a beaucoup de motifs répétitifs dans cette matrice ce qui suggère une forte corrélation entre les variables.
Nous observons qu'il y a à la fois beaucoup de blocs dont la valeur est proche de 1 montrant qu'il y a bien une corrélation entre certaines variables. Cependant, nous observons aussi qu'il y a certains blocs bleu dont la valeur est proche de 0.2 ce qui montre qu'il y a aussi certaines variables qui semblent moins corrélées.
Finalement, nous pouvons conclure qu'il semblerait y avoir beaucoup de corrélations entre les variables du dataset.

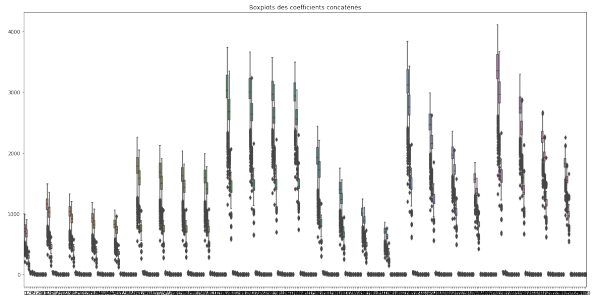
La figure 4b présente le boxplot des coefficients de scattering pour les ordres 0, 1 et 2. Nous pouvons remarquer la présence de quelques outliers, et observer que ces variables ne sont pas centrées, avec une variance relativement élevée. Cela s'explique

par le fait que l'ordre 0 représente une moyenne, tandis que les ordres 1 et 2 capturent respectivement les variations locales et les corrélations entre ces variations. Ainsi, il est essentiel de centrer et de réduire nos données avant d'entraîner tout modèle, afin d'améliorer la performance et la robustesse des résultats.

Enfin, la figure 5 représente la projection des coefficients de scattering sur les deux premières composantes principales après avoir effectué une ACP sur nos coefficients de scattering.
Nous remarquons que l'axe 1 capture la plupart de l'information avec 55% de la variance expliquée. L'axe 2 quant à lui explique 23% de la variance.
Avec 3 composantes nous captons environ 98% de la variance, chacun des axes pourrait représenter les ordre 0, 1 et 2 du scattering 3D.



(a) Corrélation des coefficients de scattering



(b) Boxplot des coefficients de scattering

FIGURE 4 – Quelques analyses de données sur les coefficients de scattering

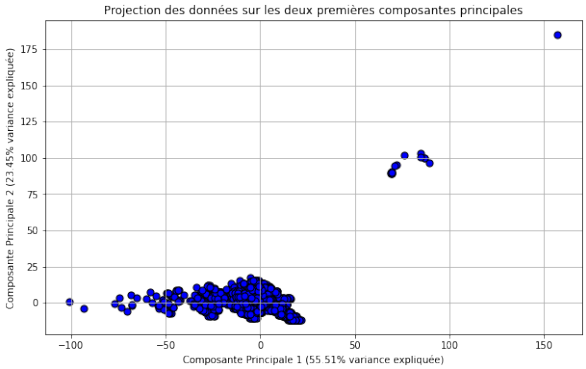


FIGURE 5 – Résultats de la projection des deux premières composantes principales de l'ACP sur les coefficients de scattering

4.2 Modèle

Comme expliqué précédemment nous avons vu qu'il y a une grande corrélation entre certaines variables. Cette corrélation pourrait être le signe d'une relation linéaire entre les variables.
Nous avons donc d'abord vérifier cette hypothèse en testant d'appliquer des modèles de régression linéaire sur nos coefficients de scattering.

Avant d'appliquer un modèle sur nos données nous centrons et réduisons nos données car comme vu sur la figure 4b les coefficients n'ont pas le même ordre de grandeur.

Nous avons ensuite testé différents modèles de régression linéaire comme la régression PLS, Ridge, Lasso, et les méthodes Multi-Layer Perceptron ou Support Vector Regression.

Pour chaque méthode nous avons réalisé un GridSearch sur chaque paramètre de chaque méthode dont l'objectif est de **minimiser la Mean Absolute Error (MAE)**.

Voici une liste des paramètres testés pour chaque méthode :

— PLS : n_components (nombre de composantes à garder).

— Ridge : alpha (régularisation), solveur, nombre d'itérations maximal, intercept, tolérance.

— Lasso : alpha, intercept, nombre d'itérations maximal, tolérance.

— Support Vector Regression : noyau, degré, gamma, tolérance, C (régularisation), epsilon.

— Multi-Layer Perceptron : taille des couches cachées, solver, alpha.

Chacune de ces méthodes a été testées sur les paramètres de la grille suivante :

— Hyperparamètres de la grille : M, N, O = 192, 128, 96.

— Hyperparamètres du Scattering3D : J = 2, L=3

4.3 Résultats

Le tableau 1 présente les résultats obtenus avec différentes méthodes présentées dans la partie précédente.

Méthode	Ridge	Régression PLS	Ridge + Moyenne	Lasso	MLP	SVR
Score	0.003231 / 0.0018	0.016	3.6	0.11	2.18	2.88

TABLE 1 – Résultats obtenus avec différentes méthodes de régression

Les meilleurs résultats ont été obtenus avec la méthode Ridge en modifiant les paramètres du scattering.
Nous avons sélectionné **M, N, O = 200, 150, 100**, **J, L= 5,4**.
Nous avons utilisé un terme de régularisation de $1e-9$ avec le solveur Cholesky.
Nous obtenons un score de **0.0018** sur tout le dataset de test.
Ce résultat est bien meilleur qu'en conservant les paramètres de la grille initiaux car en prenant J et L plus grand nous allons garder plus d'informations lors du scattering.

4.4 Autres méthodes testées

Pour mettre en avant l'invariance des coefficients de scattering nous avons appliqué la moyenne sur la première dimension et nous avons appliqué le modèle de régression ridge.

Cette méthode nous a donné un score de 3.6, ce qui est moins bien que ce que nous avions précédemment.

5 Pistes d'amélioration

Pour améliorer nos résultats, nous avons réalisé un état de l'art des méthodes les plus récentes. Nous avons conclu que l'utilisation des Graph Neural Networks (GNN) [5] pourrait être une approche intéressante. Nos données étant constituées des positions des atomes, nous pouvons représenter les molécules sous forme de graphes. Cette représentation permet d'extraire un maximum de caractéristiques à l'aide des couches d'un réseau GNN, optimisant ainsi la capacité d'estimer l'énergie moléculaire.

En ce qui concerne la matrice de Coulomb, les KAN (KAN : Kolmogorov–Arnold Networks) [4] peuvent être une piste pour améliorer les résultats. En effet, cette méthode dépasse les MLP en terme de performances sur des tâches de régression. Le but est d'apprendre les activations au lieu des poids du réseau.

Malheureusement, nous avons pas eu le temps de tester c'est deux méthodes.

6 Conclusion

Pour conclure, ce projet nous a permis de découvrir de nouvelles méthodes, notamment le 3D scattering. Bien que cette méthode prenne beaucoup de temps pour calculer les coefficients de scattering, elle semble très performante pour capturer un maximum d'informations et de caractéristiques dans une représentation invariante aux rotations et translations. Cela nous a permis d'obtenir un score assez bon dans la compétition Kaggle. Cependant, ce score peut être amélioré en testant des méthodes de recherche plus récentes et prometteuses.

Références

[1] 3d scattering quantum chemistry regression. https://www.kymat.io/gallery_3d/scattering3d_qm7_torch.html#sphx-glr-gallery-3d-scattering3d-qm7-torch-py.

[2] S.Fazli1 M.Rupp G.Montavon, K.Hansen. Learning invariant representations of molecules for atomization energy prediction. 2012.

[3] M.Hirn S.Mallat L.Thiry M.Eickenberg, G.Exarchakis. Solid harmonic wavelet scattering for predictions of molecule properties. *physics.chem-p*, 2018.

[4] S.Vaidya F.Ruehle J.Halverson M. Soljacic T.Hou M.Tegmark Z.Liu, Y.Wang. Kan : Kolmogorov–arnold networks. 2024.

[5] H.Wang C.Lu C.Lee Z.ZHANG, Q.Liu. Motif-based graph self-supervised learning for molecular property prediction properties. 2021.