```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Set universal theme for figures
plt.style.use('seaborn')

# Configure plot settings
plt.rcParams['figure.figsize'] = (6, 4.5)
plt.rcParams['figure.dpi'] = 300
plt.rcParams['axes.grid'] = True
# Set numerical precision
np.set_printoptions(precision=5)
```

Links: Google Image: image Bold Text: **Ram** Italic Text: *Krishna* Code Chunk:

```
import os
```

Guides for Writing your report

For the full document:

- you can use a html output but you need to keep the sectioning
- You are writing a comprehensive report
- The goal is to communicate to others
- Need to focus on communicating
- There should be no code or text not in the appropriate headings below
- Delete/comment out extra notes (like these) or others so your document is as clear as it can be
- Reference your figures, tables, equations in the document.
- Captions are needed on all figures, tables, and equations.
- Think about your audience and clearly communicating.

For the updates:

- please include the previous update for each Submitted Update Document
  - Each update is important to keep for grading
- For the final report remove updates and give a full comprehensive report
- Delete or comment out my notes that you are not using when you submit your documents
  - The extra clutter makes communication difficult

Update 2

- Please put a bulleted list of things you have accomplished since the last update

    - Include things that didn't work but you tried
    - Things you are planning on doing
    - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

Update 1

- Please put a bulleted list of things you have accomplished since the last update
    - Include things that didn't work but you tried
    - Things you are planning on doing
    - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

Excuetive Summary

- Summarize the key (This could be a bulleted list)
    - information about your data set
    - major data cleaning
    - findings from EDA
    - Model output
    - Overall conclusions

Abstract

The accurate prediction of molecular properties from structural information is essential for accelerating discovery in chemistry and materials science. While Density Functional Theory (DFT) provides reliable quantum mechanical predictions, its high computational cost limits its applicability in large-scale screening. In this study, we develop a neural network-based regression model to predict molecular properties—specifically total energy—directly from three-dimensional atomic coordinates. Using the **DFT_all.npz** dataset available from Zenodo, which contains a variety of DFT-computed properties for small organic molecules, we train the model in a supervised manner to learn the structure–property relationship. Our results demonstrate that neural networks can effectively approximate DFT-level accuracy while significantly reducing computation time. This work highlights the potential of machine learning as a scalable alternative to traditional quantum chemical simulations, enabling faster exploration of chemical space for materials and drug design.

Introduction

Predicting molecular properties directly from structural information is a fundamental task in computational chemistry and materials science. Traditionally, this is achieved through quantum mechanical methods such as Density Functional Theory (DFT), which provide accurate predictions but are computationally expensive and limited in scalability. As the demand grows for rapid property evaluation in high-throughput screening and molecular design, data-driven alternatives have gained significant attention.

Recent advances in machine learning, particularly neural networks, have opened new pathways for modeling the complex relationship between a molecule's structure and its physicochemical properties. These models can learn from large datasets of precomputed molecular structures and properties to make fast, accurate predictions without relying on costly simulations.

In this project, we focus on **predicting molecular properties from 3D molecular structures** using supervised learning with neural networks. We use the **DFT_all.npz** dataset, derived from DFT calculations and available through Zenodo, which contains atomic coordinates and quantum-level properties for a variety of small organic molecules.

Our goal is to **train a neural network to accurately predict key molecular properties—such as total energy—from 3D atomic coordinates**, thereby capturing the structure–property relationship encoded in quantum mechanical simulations. This approach aims to demonstrate how machine learning models can serve as efficient surrogates for DFT, accelerating materials discovery and molecular design through predictive modeling.

Data Science Methods

We decided to realize a Property prediction of the elements from their 3D molecular structure. We will use Supervising trqining on Neural Network. To do so we will use the 3D properties of the molecules as training inputs, then we will train the network with the use of the desired Property (Atomization Energy for example) as a label.

The input data should need no pretreatment. The output label should be a continous value defining the property of the element.

The dataset is composed of 784875 element wich is a quite huge amount of data (to compare, MNIST which is a basic digit recognition dataset contains 70000 elements) so the split between training subset and test subset should be relevant.

The main limit of this method is for each property we would lik to predict, we would have to entirely redo the training with a different label.

Exploratory Data Analysis

Explanation of your data set

- How many variables?
  - 784875 data elements described by 26
- What are the data classes?
  - compounds : dtype = array
  - atoms : dtype = array
  - freqs:dtype = array
  - vibmodes:dtype = array
  - zpves:dtype = float64
  - U0:dtype = float64
  - U298:dtype = float64

  - H:dtype = float64
  - S:dtype = float64
  - G:dtype = float64
  - Cv:dtype = float64
  - Cp:dtype = float64
  - coordinates:dtype = array
  - Vesp:dtype = array
  - Qmulliken:dtype = array
  - dipole:dtype = array
  - quadrupole:dtype = array
  - octupole: dtype = array
  - hexadecapole:dtype = array
  - rots:dtype = array
  - gap: dtype = float64
  - Eee: dtype = float64
  - Exc:dtype = float64
  - Edisp: dtype = float64
  - Etot:dtype = float64
  - Eatomization: dtype = float64
- How many levels of factors for factor variables?
  - atoms$\to$ level 10.
  - compounds$\to$ level 784837.
- Is your data suitable for a project analysis?
  - Yes, we think. Sufficient variables are included in this dataset.
- Write you databook, defining variables, units and structures

variables	units	discreption
compounds		Stoichiometric formulas of the molecules
atoms		Atomic numbers in the molecule
freqs	cm^{-1}	Vibrational frequencies obtained from harmonic frequency analysis.
vibmodes	$\r{A}$	Normal modes of vibration represented as displacement vectors.
U0	Ha	Internal energy at 0 K
U298	Ha	Internal energy at 298 K
H	Ha	Enthalpy
S		Entropy
G	Ha	Gibbs free energy
Cv		Heat capacity at constant volume
Cp		Heat capacity at constant pressure
coordinates		coordinates (XYZ) of atoms in the molecule.

variables	units	discreption
Vesp		Electrostatic potential
Qmulliken		Mulliken atomic charges
dipole	a.u.	Dipole moment
quadrupole	a.u.	Quadrupole moment
octupole	a.u.	Octupole moment
hexadecapole	a.u.	Hexadecapole moment
rots	MHz	Rotational constants of the molecule.
gaps	Ha	HOMO-LUMO energy gap
Eee	Ha	Electron-electron repulsion energy
Exc	Ha	Exchange-correlation energy
Edisp	Ha	Dispersion correction energy
Etot	Ha	Total electronic energy
Eatomization	Ha	Atomization energy

Data Cleaning

- We performed data cleaning as follows:

```
 # Filter fixed-length molecules (same n_atoms)
n_atoms_arr = np.array([len(a) for a in atoms])
most_common_n = Counter(n_atoms_arr).most_common(1)[0][0]
filtered_data = [
   (coord, atom) for coord, atom in zip(coordinates, atoms)
   if len(atom) == most_common_n
]
filtered_coords = [c for c, _ in filtered_data]
filtered_atoms = [a for _, a in filtered_data]
# Construct X as 3D input: [x, y, z, Z]
X = np.array([
   np.hstack([coord, atom.reshape(-1, 1)])  # shape: (n_atoms, 4)
   for coord, atom in zip(filtered_coords, filtered_atoms)
])
# Select targets
targets = ['U0', 'H', 'gap']
Y_all = np.column_stack([data[prop] for prop in targets])
Y = np.array([
   y for y, atom in zip(Y_all, atoms) if len(atom) == most_common_n
])
```

- Discription of this code

- 1 Count the number of atoms in each molecule.
  - `python n_atoms_arr = np.array([len(a) for a in atoms])`
- 2 Find the most common atom count among all molecules.
  - `python most_common_n = Counter(n_atoms_arr).most_common(1)[0][0]`
- 3 Filter the data to keep only the molecules with that common atom count (when using a CNN, the input data must have consistent dimensions).

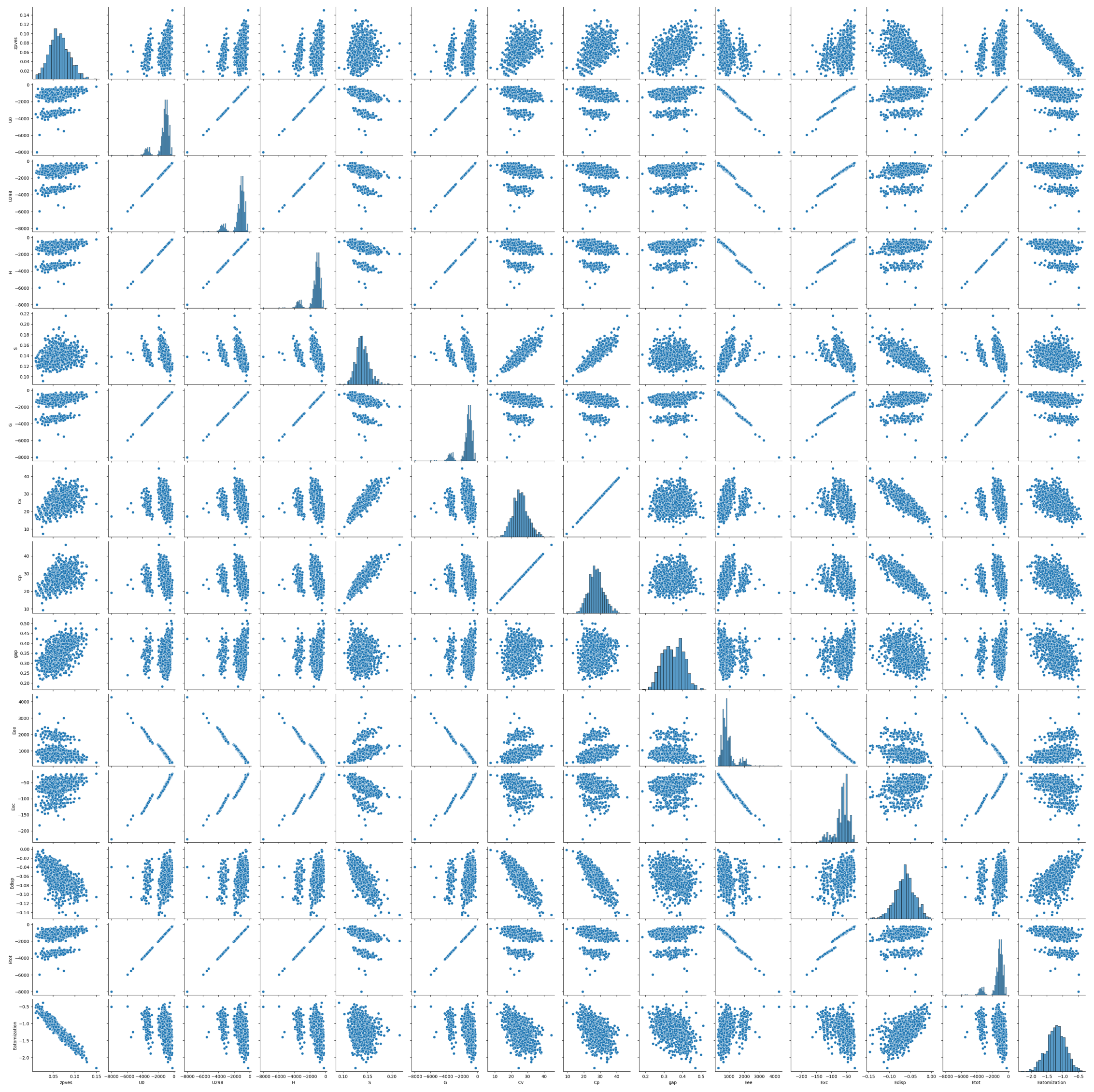
  -
    ```
    filtered_data = [
    (coord, atom) for coord, atom in zip(coordinates, atoms)
    if len(atom) == most_common_n
    ]
    filtered_coords = [c for c, _ in filtered_data]
    filtered_atoms = [a for _, a in filtered_data]
    ```
- 4 Build the input features X by stacking atomic coordinates `python[x, y, z]` with atomic numbers Z.

  -
    ```
    X = np.array([
      np.hstack([coord, atom.reshape(-1, 1)])  # shape: (n_atoms, 4)
      for coord, atom in zip(filtered_coords, filtered_atoms)
    ])
    ```
- 5 Define taeget values Y.

  -
    ```
    targets = ['U0', 'H', 'gap']
      Y_all = np.column_stack([data[prop] for prop in targets])
      Y = np.array([
      y for y, atom in zip(Y_all, atoms) if len(atom) == most_common_n
    ])```
    ```

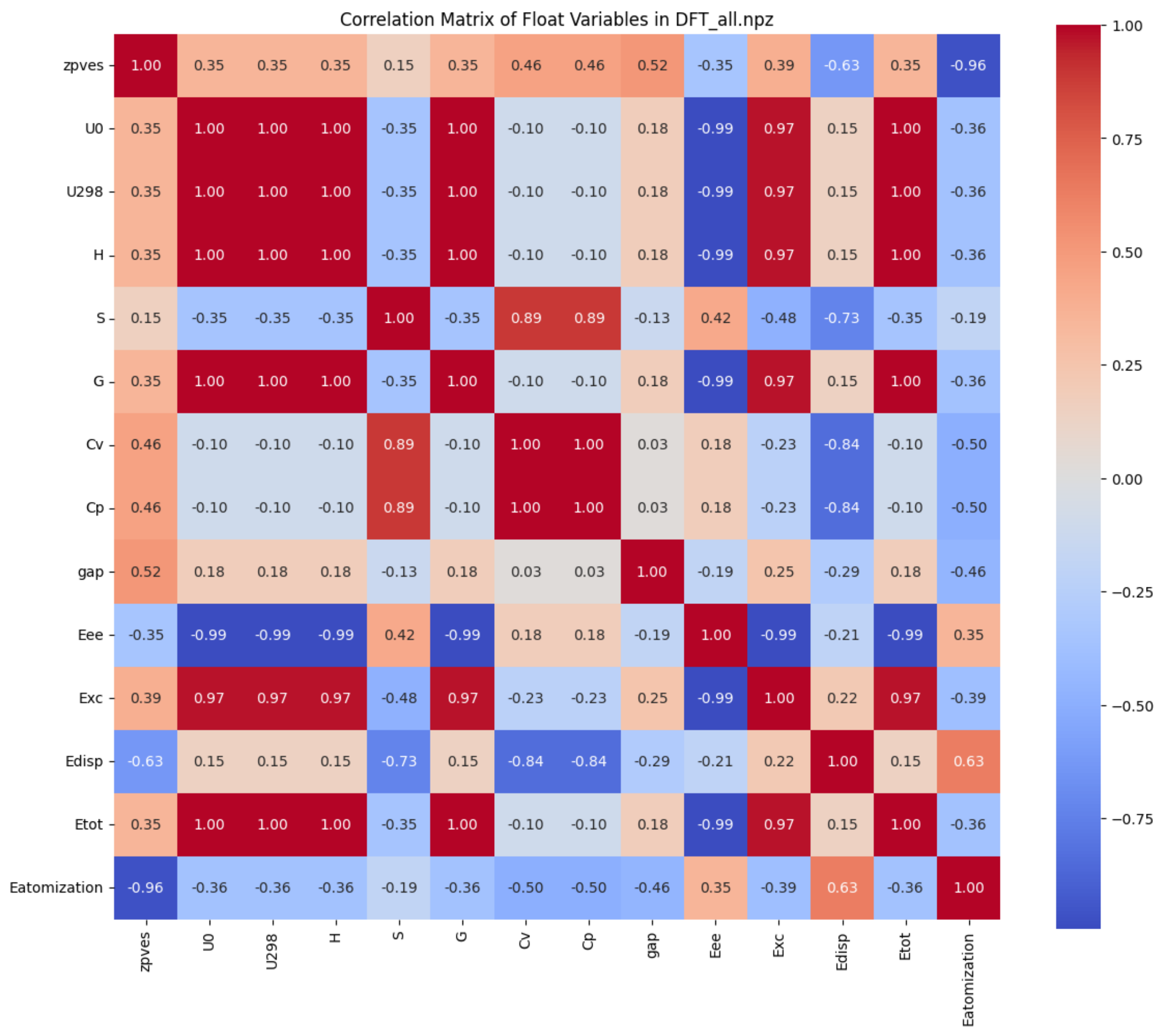
Data Vizualizations

- Here is data visualizations (DFT_all.npz):
  - variable correlations



Variable Correlations

- Here is variable correlations (DFT_all.npz):



Statistical Learning: Modeling & Prediction

- least 1 simple linear model (or simple logistic model)
- requires the appropriate modeling for your data set including machine learning
- Types of modeling to try
- Statistical prediction/modeling
- Model selection
- Cross-validation, Predictive R2
- Interpret results
- Challenge results

Discussion

- Discussion of the answers to the data science questions framed in the introduction

Conclusions

Acknowledgments

References

- Include a bib file in the markdown report
- Or hand written citations.