# Analysis of CVAE Implementation

## Enhancing Performance through Loss Function Optimization and Architecture Tuning

A.Ghanaatian   Supervisor: Prof. Caragea

November 29, 2024

# Introduction

- **Objective**: Improve the model by implementing several key enhancements:
  - Added plotting to compare real vs. predicted values.
  - Incorporated an **Energy Loss**$^2$ component to minimize energy discrepancies.
  - Included **MRE**$^2$ in the loss function for differentiability and metric minimization.
  - Fine-tune all Parameters and Hyper-Parameters to optimize the new loss function and training phase to get a better model

## Understanding Energy Components

- **Energy Calculations** are being used for the **Energy Loss** term in our loss function.
- Components involved:
    1. **Kinetic Energy (KE)**
    2. **Potential Energy (PE)**
    3. **Error Calculation between KE and PE**

# Kinetic Energy (KE)

The total kinetic energy is the sum of the kinetic energies of Carbon (C), Oxygen (O), and Sulfur (S):

$$KE = KE(C) + KE(O) + KE(S)$$

Where:

$$KE(C) = \frac{p_{Cx}^2}{2m_C} + \frac{p_{Cy}^2}{2m_C} + \frac{p_{Cz}^2}{2m_C}$$

$$KE(O) = \frac{p_{Ox}^2}{2m_O} + \frac{p_{Oy}^2}{2m_O} + \frac{p_{Oz}^2}{2m_O}$$

$$KE(S) = \frac{p_{Sx}^2}{2m_S} + \frac{p_{Sy}^2}{2m_S} + \frac{p_{Sz}^2}{2m_S}$$

Masses:

$$m_C = 21894.71361 \quad m_O = 29164.39289 \quad m_S = 58441.80487$$

# Potential Energy (PE)

The potential energy is calculated based on the distances between the atoms:

$$PE = \frac{4}{r_{CO}} + \frac{4}{r_{CS}} + \frac{4}{r_{OS}}$$

Where:

$$r_{CO} = \sqrt{(c_x - o_x)^2 + (c_y - o_y)^2 + (c_z - o_z)^2}$$
$$r_{CS} = \sqrt{(c_x - s_x)^2 + (c_y - s_y)^2 + (c_z - s_z)^2}$$
$$r_{OS} = \sqrt{(o_x - s_x)^2 + (o_y - s_y)^2 + (o_z - s_z)^2}$$

# Energy Error Calculation

The error between kinetic and potential energy is calculated as:

$$\text{Error} = \frac{|KE - PE|}{|KE|}$$

- This error is minimized in the **Energy Loss** component of our loss function.

# Comprehensive Loss Function (Part 1)

The combined **Loss Function** can be expressed as:

$$\text{Loss} = \alpha_1 \cdot \text{Energy Diff}^2 + \alpha_2 \cdot \text{MRE}^2 + \alpha_3 \cdot \text{MSE}$$

Where:

1. **Energy Difference**:

$$\text{Energy Diff} = \frac{|KE - PE|}{|KE|}$$

   $E$ is the true energy, $\hat{E}$ is the predicted energy.

2. **Mean Relative Error (MRE)**:

$$\text{MRE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|x_i - \hat{x}_i|}{|x_i|} \times 100$$

   $x_i$ and $\hat{x}_i$ are the true and predicted values.

3. **Mean Squared Error (MSE)**:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2$$

# Comprehensive Loss Function (Part 2)

Continuing the combined **Loss Function**:

$$\text{Loss} = \alpha_4 \cdot \text{KL} + \alpha_5 \cdot \text{L1} + \alpha_6 \cdot \text{L2}$$

Where:

4. **KL Divergence**:

$$KL(N(\mu, \sigma^2) \parallel N(0, 1)) = \frac{1}{2} \sum_{i=1}^{d} \left( \sigma_i^2 + \mu_i^2 - 1 - \log(\sigma_i^2) \right)$$

   $\mu_i$ and $\sigma_i^2$ are the mean and variance from the encoder for dimension $i$.

5. **L1 Regularization**:

$$\text{L1} = \sum_j |w_j|$$

   $w_j$ are the model parameters.

6. **L2 Regularization**:

$$\text{L2} = \sum_j w_j^2$$

# Loss Function Weighting Factors

- **Weighting Coefficients** ($\alpha$) are critical for balancing the loss components.
- Fine-tuning these coefficients helps in:
  - Prioritizing certain loss terms over others.
  - Achieving better convergence and model performance.

# Model Architecture Parameters

The model architecture is defined by three key factors:

1. **Hidden Dimension** (`hidden_dim`)
2. **Number of Layers** (`layer_num`)
3. **Latent Layer Size**

- **Encoder and Decoder Layers**:

$$[\text{hidden\_dim} \times 2^{i}] \quad \text{for} \quad i \in [0, \text{layer\_num})$$

## Example Model Architecture

Given the parameters:

- **Hidden Dimension** (`hidden_dim`): 16
- **Number of Layers** (`layer_num`): 3
- **Latent Layer Size**: 128

The model architecture becomes:

- **Input Layer**: 9 vectors
- **Encoder**: $16 \rightarrow 32 \rightarrow 64 \rightarrow 128$
- **Decoder**: $128 \rightarrow 64 \rightarrow 32 \rightarrow 16$
- **Output Layer**: 9 vectors

# Training Regularization

To prevent overfitting and improve generalization:

- **Regularization Terms**:
  - **L1 Regularization**:

$$\text{L1} = \sum_j |w_j|$$

  - **L2 Regularization**:

$$\text{L2} = \sum_j w_j^2$$

# Training Hyperparameters

Key hyperparameters during training:

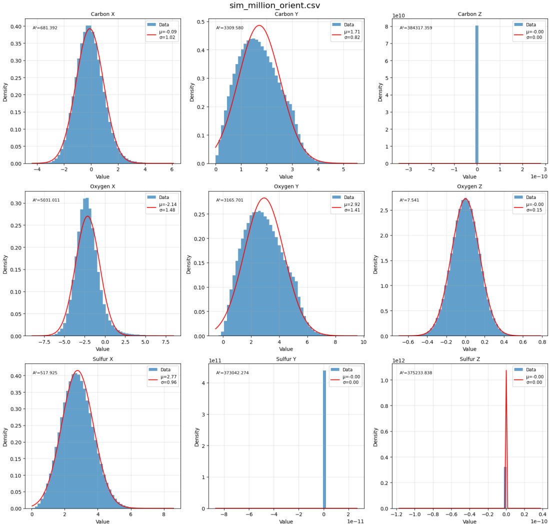
- **Batch Size**
- **Learning Rate**
- **Number of Epochs**

To enhance efficiency:

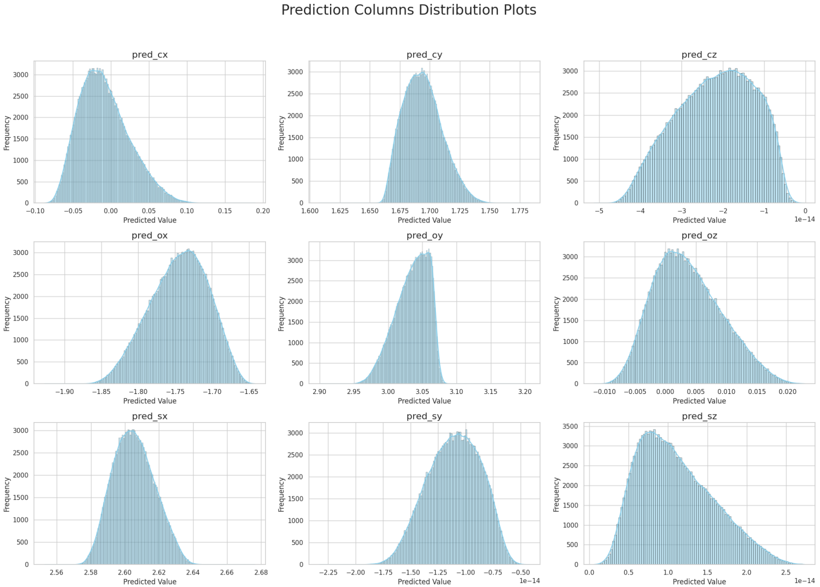
- Implemented **Early Stopping** with:
  - **Patience Steps**
  - **Minimum Delta** (`min_delta`)

# Test Set Evaluation

- **Test Set**: 15% of the original 1 million-row dataset.



# Test Set Prediction Plot



# Performance Comparison

## Model Performance Metrics

| **Model** | **MRE (%)** | **Energy Loss** | **MSE** |
|---|---|---|---|
| **Random** | 85.38 | $3.24 \times 10^{-1}$ | 0.17 |
| **Sim** | 82.67 | $4.73 \times 10^{-1}$ | 0.78 |

## Fine-Tuning Steps, Future steps

Sequential steps taken to fine-tune the model:

1. **Adjusted Loss Function Weights**:
   - Fine-tuned the weighting coefficients $\alpha_1, \alpha_2, \ldots, \alpha_6$.
2. **Optimized Training Hyperparameters**:
   - Tweaked batch size, learning rate, and increased epochs.
   - Removed early stopping to allow full training cycles.
3. **Refined Regularization Factors**:
   - Adjusted L1, L2 regularization strengths, and Beta.
4. **Enhanced Early Stopping Parameters**:
   - Modified patience and `min_delta` for better convergence.
5. **Explored Model Architecture**:
   - Tested different configurations of layers and dimensions.