Implementation of Conditional Variational Autoencoders for Atomic Position Prediction

Technical Report

November 23, 2024

Abstract

This report presents a comprehensive implementation of Conditional Variational Autoencoders (CVAEs) for predicting initial atomic positions from post-explosion momenta. The study focuses on a three-atom system (Carbon, Oxygen, and Sulfur) in three-dimensional space, featuring flexible hyperparameter configuration, customizable activation functions, and invertible normalization methods. The implementation includes detailed energy calculations, multiple evaluation metrics, and specific visualization requirements.

1 Introduction

1.1 Background

Conditional Variational Autoencoders extend the VAE architecture by incorporating conditional inputs for controlled data generation. This implementation focuses on solving an inverse problem in atomic physics, predicting initial positions from momentum data.

2 Data Structure and Processing

2.1 Input Data

The data is imported from CSV format:

```
data = pd.read_csv(FILEPATH)
position = data[['cx', 'cy', 'cz', 'ox', 'oy', 'oz',
                 'sx', 'sy', 'sz']].values
momenta = data[['pcx', 'pcy', 'pcz', 'pox', 'poy', 'poz',
                'psx', 'psy', 'psz']].values
```

2.2 Data Split

- Training Set: 70%
- Validation Set: 15%
- Test Set: 15%

3 Implementation Requirements

3.1 Hyperparameter Configuration

All parameters must be configurable at the start of the code:

3.1.1 Hidden Layer Architecture

- Hidden dimension size: User-defined initial size
- Number of hidden layers: User-defined
- Layer sizes follow the pattern: $[\text{hidden\_dim} \cdot 2^i]$ for $i \in [0, \text{num\_layers})$
- Example: For $\text{hidden\_dim} = 64$ and $\text{num\_layers} = 3$:
  - Layer 1: 64
  - Layer 2: 128
  - Layer 3: 256

3.1.2 Training Configuration

- Early stopping with configurable:
  - Patience parameter
  - Minimum delta factor
- Flexible learning rate
- Adam optimizer
- Customizable activation functions
- Optional normalization methods (must be invertible)
- Option for no normalization

4 Model Architecture

4.1 Loss Function Components

Total loss combines multiple terms:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{reconstruction}} + \beta \cdot \text{KL}_{\text{divergence}} + \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2 \tag{1}$$

where:

- L1 (Lasso): $\mathcal{L}_1 = \sum |w|$
- L2 (Ridge): $\mathcal{L}_2 = \sum w^2$
- β-VAE term: Controls reconstruction quality vs. disentanglement
- $\beta = 1$ gives standard VAE behavior

5 Training Process

5.1 Learning Curve Requirements

- Two separate learning curves required:
  - First 10 epochs
  - Remaining epochs
- Plot based on training loss without regularization
- Validation curve on same scale as training

5.2 Data Leakage Prevention

- Test/validation phases:
  - No position data input
  - Sample only from training latent distribution
  - Use mean and standard deviation from training
- Decoder uses sampled latent variables with test momenta

6 Evaluation Metrics

6.1 Position Metrics

6.1.1 Mean Relative Error (MRE)

$$\text{MRE} = \frac{|\text{real} - \text{predicted}|}{|\text{real}| + \epsilon} \times 100\% \tag{2}$$

where $\epsilon = 10^{-10}$

6.1.2 Squared Mean Relative Error

$$\text{MRE}^2 = \left(\frac{\text{real} - \text{predicted}}{\text{real}}\right)^2 \times 100\% \tag{3}$$

6.2 Energy Calculations

6.2.1 Kinetic Energy

For each particle $i \in \{\text{C}, \text{O}, \text{S}\}$:

$$\text{KE}_i = \sum_{j \in \{x,y,z\}} \frac{p_{ij}^2}{2m_i} \tag{4}$$

Total Kinetic Energy:

$$\text{KE}_{\text{total}} = \text{KE}_\text{C} + \text{KE}_\text{O} + \text{KE}_\text{S} \tag{5}$$

6.2.2 Potential Energy

$$\text{PE} = \frac{4}{r_{\text{CO}}} + \frac{4}{r_{\text{CS}}} + \frac{4}{r_{\text{OS}}} \tag{6}$$

where distances are calculated as:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \tag{7}$$

6.2.3 Energy Difference Metrics

$$\text{EnergyDiff}_1 = \frac{|\text{KE} - \text{PE}|}{|\text{KE}|} \tag{8}$$

$$\text{EnergyDiff}_2 = \left(\frac{\text{KE} - \text{PE}}{\text{KE}}\right)^2 \tag{9}$$

7 Output Requirements

- All metrics must use original position scales (not normalized)
- Print all evaluation metrics
- Display random samples from test phase showing:
  - Real positions
  - Predicted positions

8 System Constants

8.1 Mass Values

Atom	Mass
Carbon (C)	21,894.71361
Oxygen (O)	29,164.39289
Sulfur (S)	58,441.80487

Table 1: Atomic mass values used in energy calculations

9 Example Results

9.1 Position Predictions

Atom	Coordinate	Real	Predicted	MRE (%)
Carbon	x	1.2801	1.3000	1.55
	y	1.8885	1.9000	0.61
	z	-0.0644	-0.0600	6.83
Oxygen	x	-0.6929	-0.7000	1.02
	y	3.0627	3.0500	0.41
	z	2.22E-16	1.00E-16	54.95
Sulfur	x	4.1056	4.0900	0.38
	y	8.48E-16	9.00E-16	6.13
	z	3.14E-16	3.50E-16	11.46

Table 2: Sample prediction results with corresponding MRE values

Average MRE across all coordinates: 9.26%

9.2 Energy Results

For the sample case, the energy components are as follows:

- Total Kinetic Energy (KE_{total}) $\approx$ 2001.32847
- Total Potential Energy (PE_{total}) $\approx$ 2001.32847
- Energy Difference (ΔE) $\approx 2.3 \times 10^{-7}$

10 Conclusion

This implementation provides a flexible and comprehensive CVAE solution for atomic position prediction, with configurable architecture, robust evaluation metrics, and specific visualization requirements. The system maintains data integrity through careful leakage prevention and supports various activation functions and normalization methods while ensuring accurate energy calculations and position predictions.