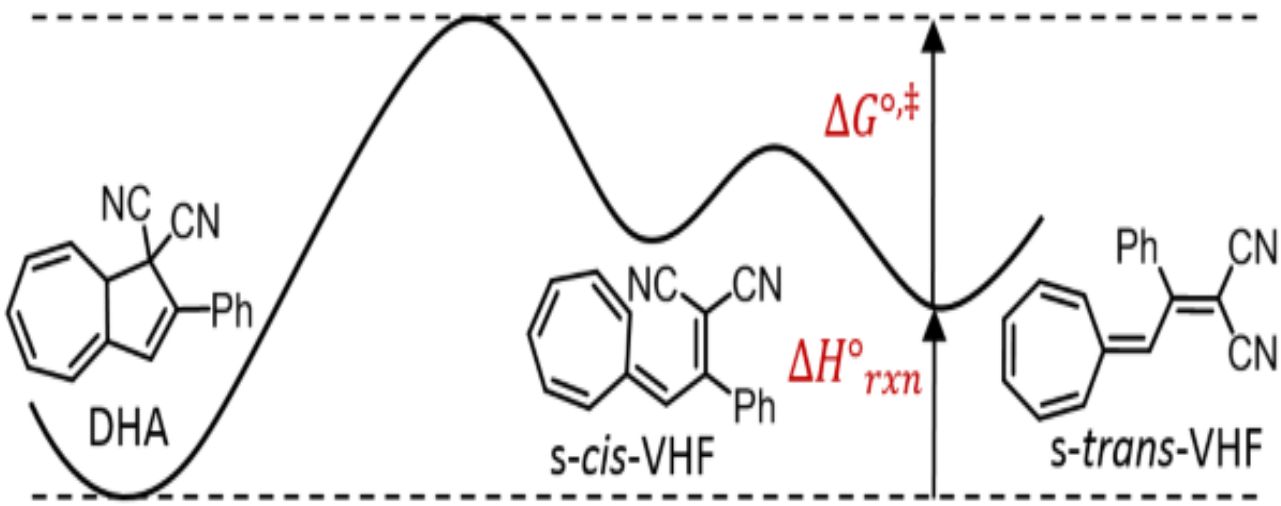
Machine Learning on Molecular Photoswitch Database

Optimizing and predicting properties of molecular solar heat batteries

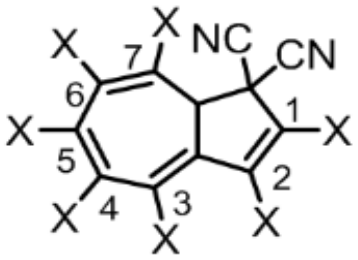
Outline

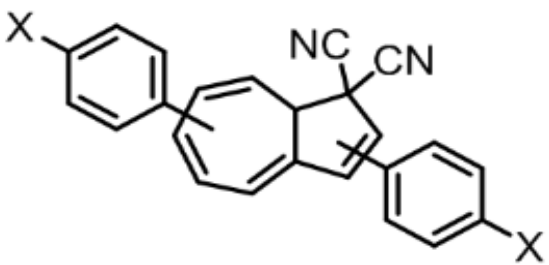
- Introduction and Motivation
- Approach
  - One-Hot Encoding
  - Convolutional Neural Network
  - XGBoost (Gradient Boosted Random Forest)
- Summary
- Outlook
- Appendix



Data acquisition

- Considered 7 different positions and 42 substituents for singly and doubly substituted systems.
- 35.588 systems / rows – 32.623 converged.
- Over 200.000 simulations.
- 53 features extracted - energy and multipoles





EWG	EDG
-[F, Cl, Br]	-OH
$-CF_3$	-OMe
-CN	$-NH_2$
$-NO_2$	$-NMe_2$
-CHO	-Me
$-CO_2H$	-NHC(O)Me
-C(O)Me	-SMe
$-C(O)NH_2$	
-CCH	
$-SO_2Me$	
-CH=NH	

What we want to do

- Can we predict the properties of a DHA/VHF derivative based only on its substituents and their position?
- Can we predict non-trivial TS properties based on DHA/VHF properties?
- Evaluate performance based on minimizing loss function, accuracy and computation time

Handling the molecule structure

- Substituent position + type (“gene”) represented by positional one-hot encoding (7*42 matrix)

Raw data

gene
0-0-2-0-31-0-0
0-0-2-0-0-31-0
0-0-2-0-0-0-31
31-0-0-2-0-0-0
0-31-0-2-0-0-0
0-0-31-2-0-0-0
0-0-0-2-31-0-0
0-0-0-2-0-31-0
0-0-0-2-0-0-31
31-0-0-0-2-0-0
0-31-0-0-2-0-0

Flattened, one-hot encoded gene

```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0])
```

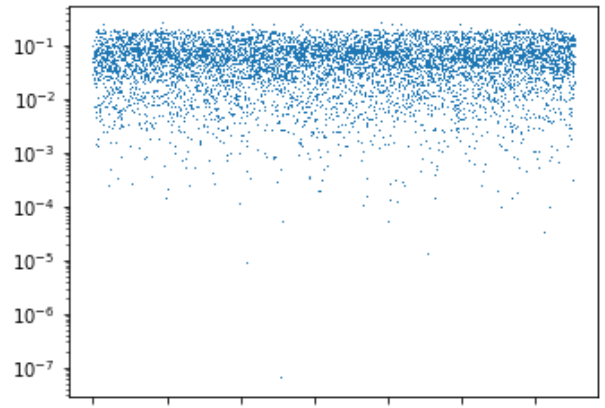
First attempt – Keras Dense Neural Network

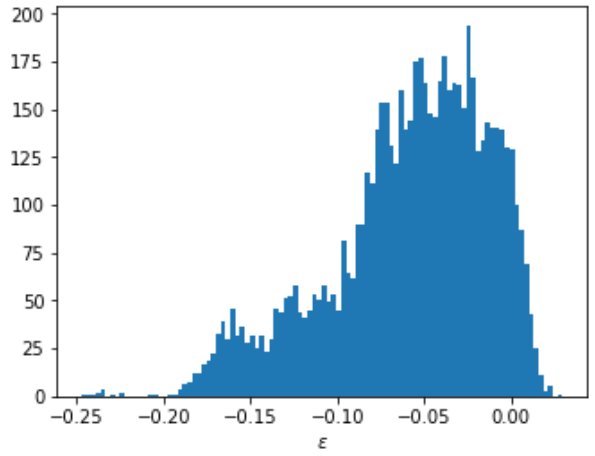
- Define Early Stopping conditions to avoid overtraining

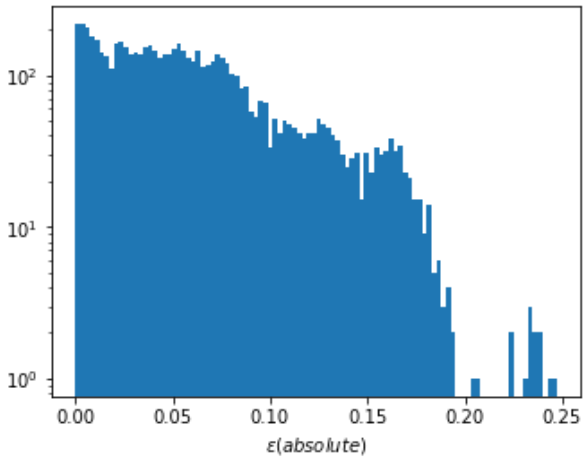
```
def DenseNN():
    model = Sequential()

    model.add(Dense(units=294, activation='relu', input_dim=294))
    model.add(Dense(units=256, activation='relu'))
    model.add(Dense(units=128, activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(units=64, activation='relu'))
    model.add(Dense(units=16, activation='relu'))
    model.add(Dense(units=1))
    model.compile(loss='logvosh', optimizer='Nadam')

    return model
```

```
esNOSTOP = EarlyStopping(monitor='val_loss', mode='min', patience=20, restore_best_weights=True)
reduce_lr = keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5,
                                              mode='auto', min_delta=0.0001, cooldown=0, min_lr=0)
```

DenseNN – DHA_E_XTB Distribution of errors



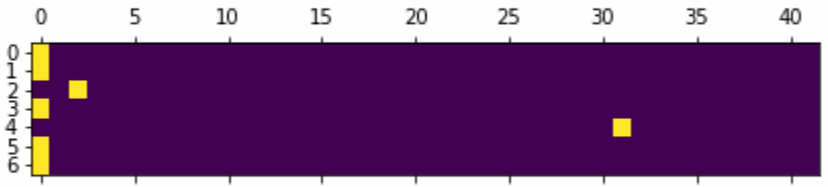




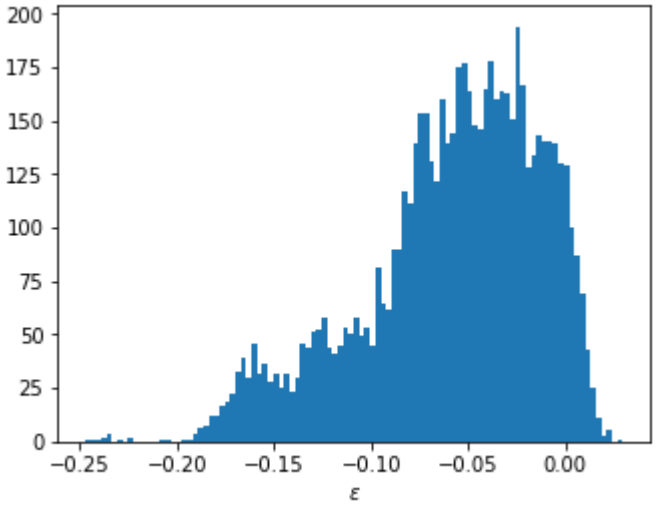
Relative error

Convolutional NN – Treating the gene as an image

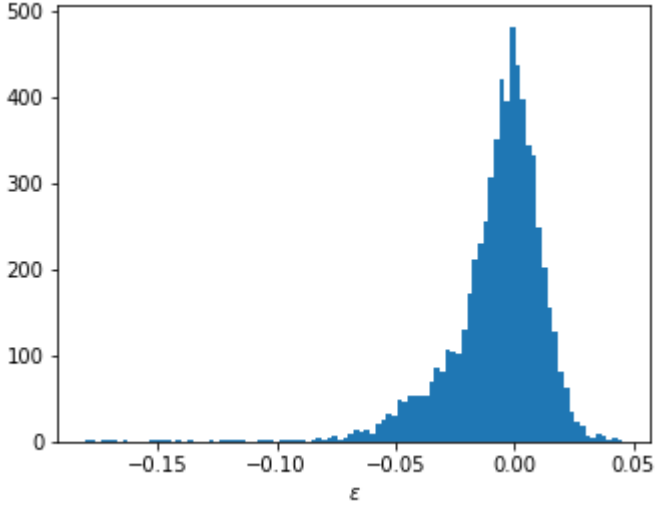
```python
def ConvNN(filters=64, kernel_size=7,
           unit1=256, unit2=128, unit3=64, unit4=16,
           dropout1=0.2, dropout2=0.2, dropout3=0.2):
    model = Sequential()
    model.add(Conv2D(filters=filters,
                     kernel_size=kernel_size,
                     activation='relu', input_shape=(7, 42, 1) ))
    model.add(Flatten())

    model.add(Dense(units=unit1, activation='relu'))
    model.add(Dropout(dropout1)) ##
    model.add(Dense(units=unit2, activation='relu'))
    model.add(Dropout(dropout2))
    model.add(Dense(units=unit3, activation='relu'))
    model.add(Dropout(dropout3)) ##
    model.add(Dense(units=unit4, activation='relu'))
    model.add(Dense(units=1))

    model.compile(loss='logcosh',
                  optimizer='Nadam', metrics=['mean_absolute_error'])
    return model
```



DenseNN



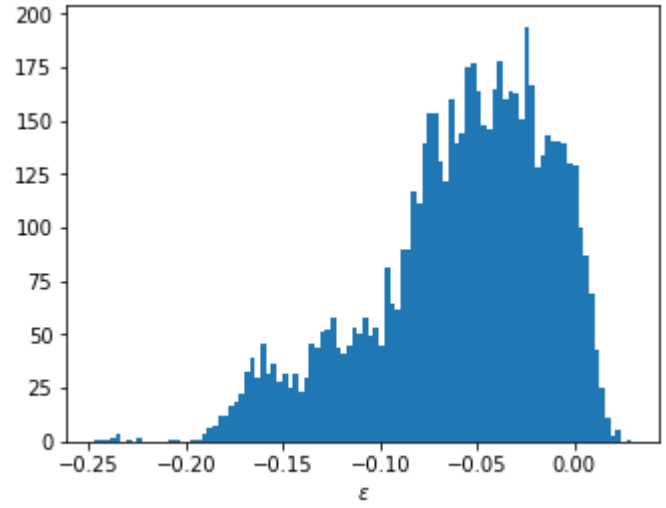
ConvNN



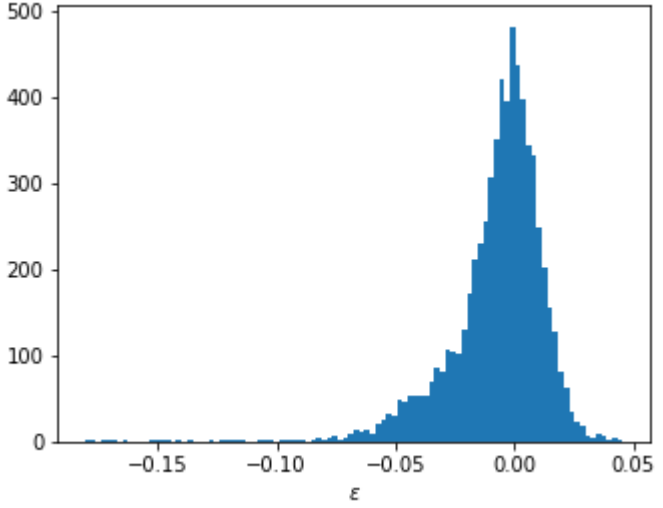
Convolutional NN – Treating the gene as an image

```python
def ConvNN(filters=64, kernel_size=7,
           unit1=256, unit2=128, unit3=64, unit4=16,
           dropout1=0.2, dropout2=0.2, dropout3=0.2):
    model = Sequential()
    model.add(Conv2D(filters=filters,
                     kernel_size=kernel_size,
                     activation='relu', input_shape=(7, 42, 1) ))
    model.add(Flatten())

    model.add(Dense(units=unit1, activation='relu'))
    model.add(Dropout(dropout1)) ##
    model.add(Dense(units=unit2, activation='relu'))
    model.add(Dropout(dropout2))
    model.add(Dense(units=unit3, activation='relu'))
    model.add(Dropout(dropout3)) ##
    model.add(Dense(units=unit4, activation='relu'))
    model.add(Dense(units=1))

    model.compile(loss='logcosh',
                  optimizer='Nadam', metrics=['mean_absolute_error'])
    return model
```

DenseNN

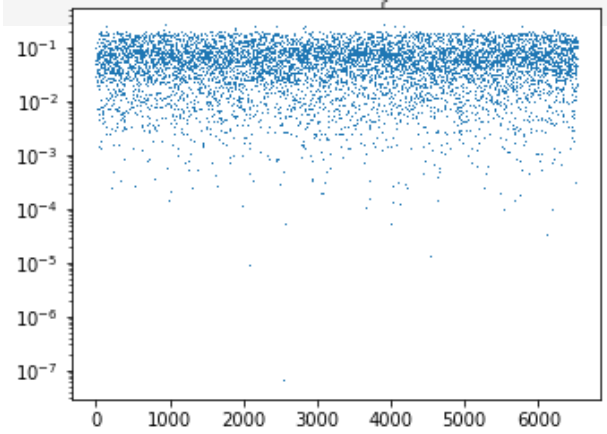


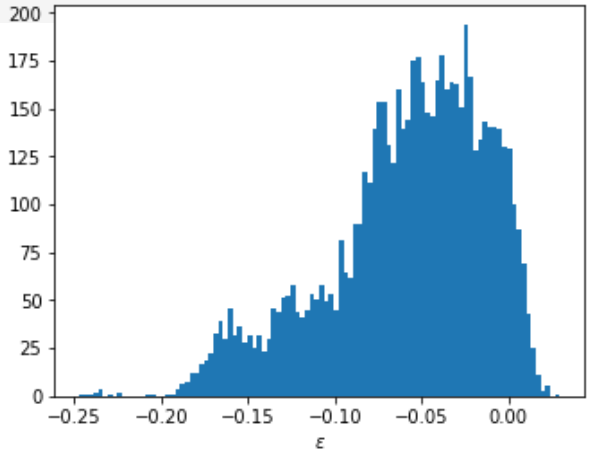
ConvNN

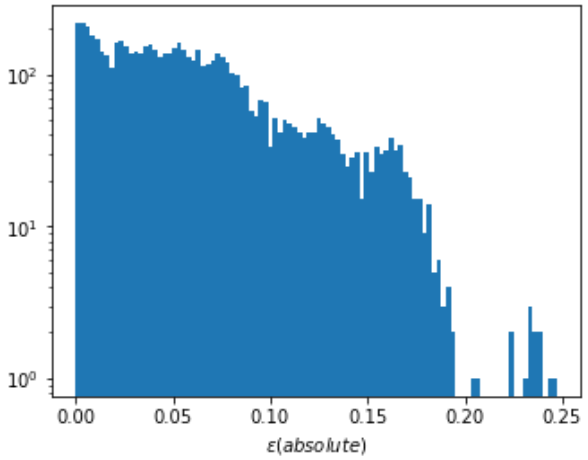


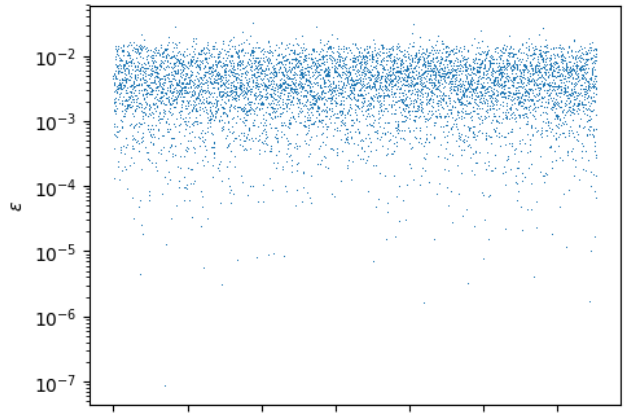
Before and after hyperparameter optimization

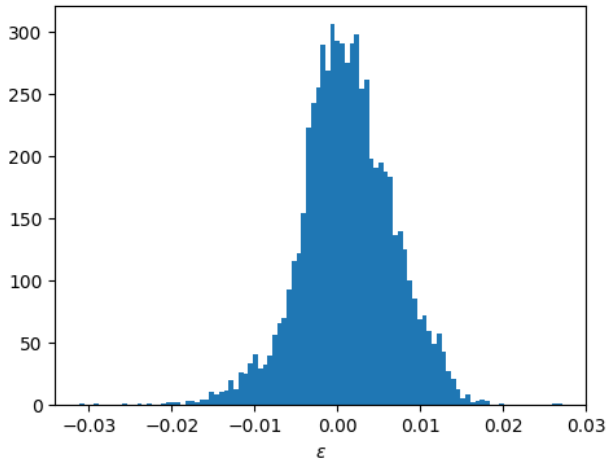
```
parameters_RandomSearch = {'filters': np.arange(2,100),
                           'kernel_size': [(1,1),(2,2),(3,3),(4,4),(5,5),(6,6),(7,7)],
                           'unit1' : np.arange(2,500),
                           'unit2' : np.arange(2,500),
                           'unit3' : np.arange(2,500),
                           'unit4' : np.arange(2,500),
                           'dropout1' : np.linspace(0.0, 1.0, 50),
                           'dropout2' : np.linspace(0.0, 1.0, 50),
                           'dropout3' : np.linspace(0.0, 1.0, 50),
                          }
```

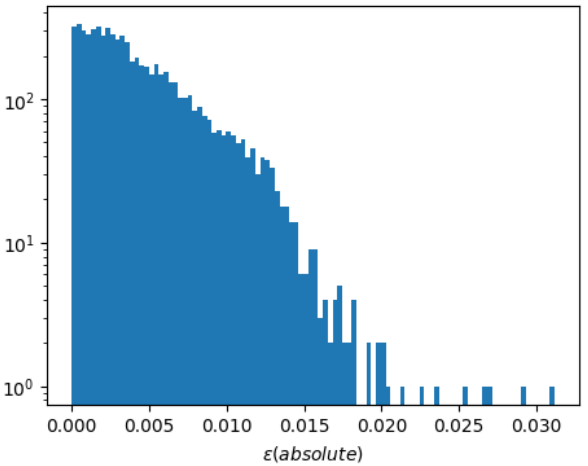


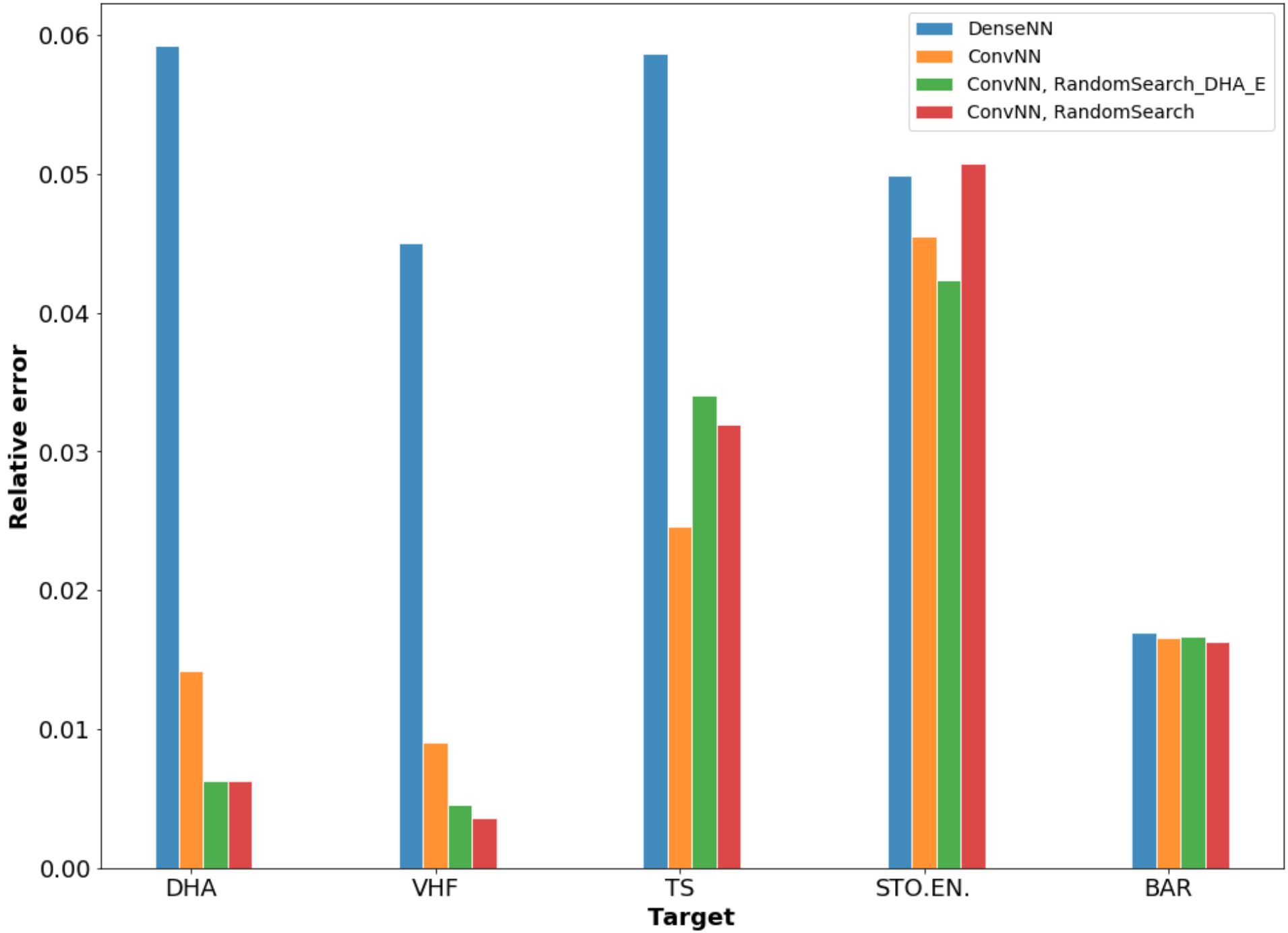




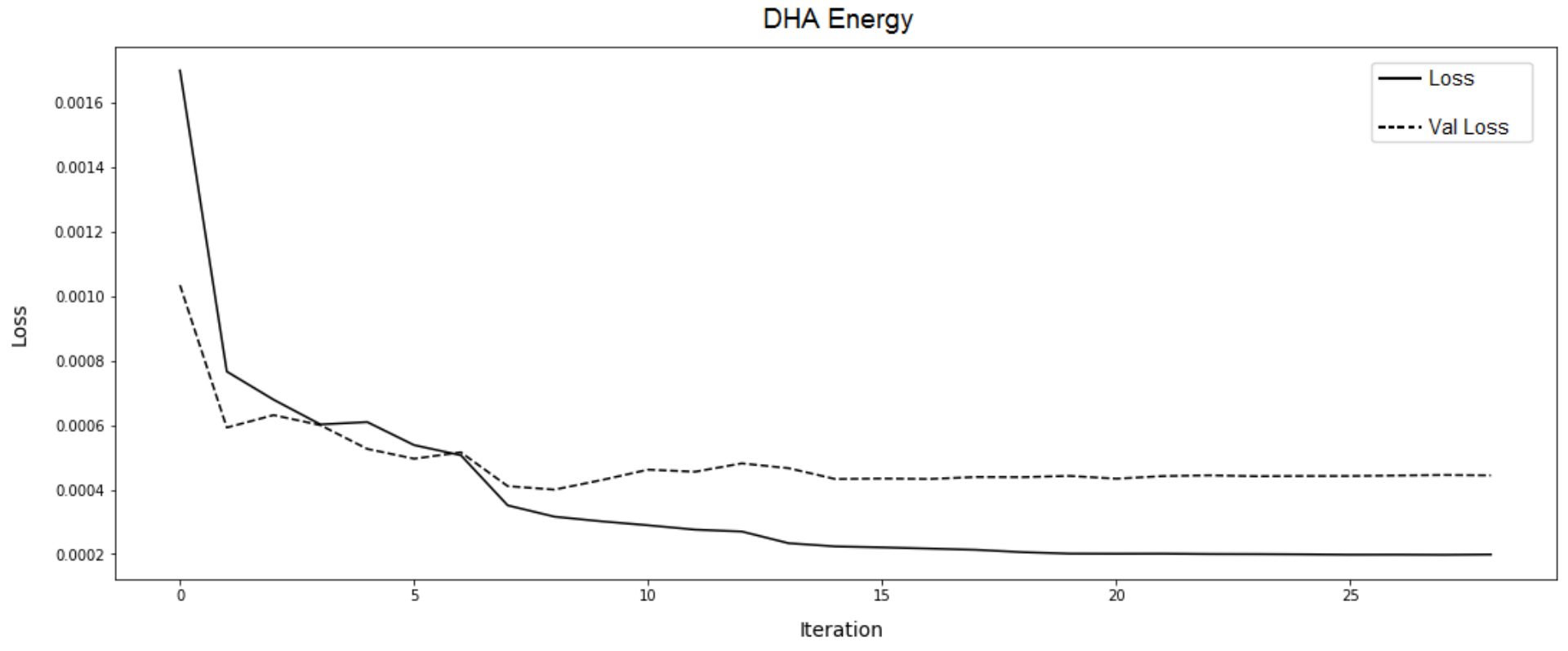




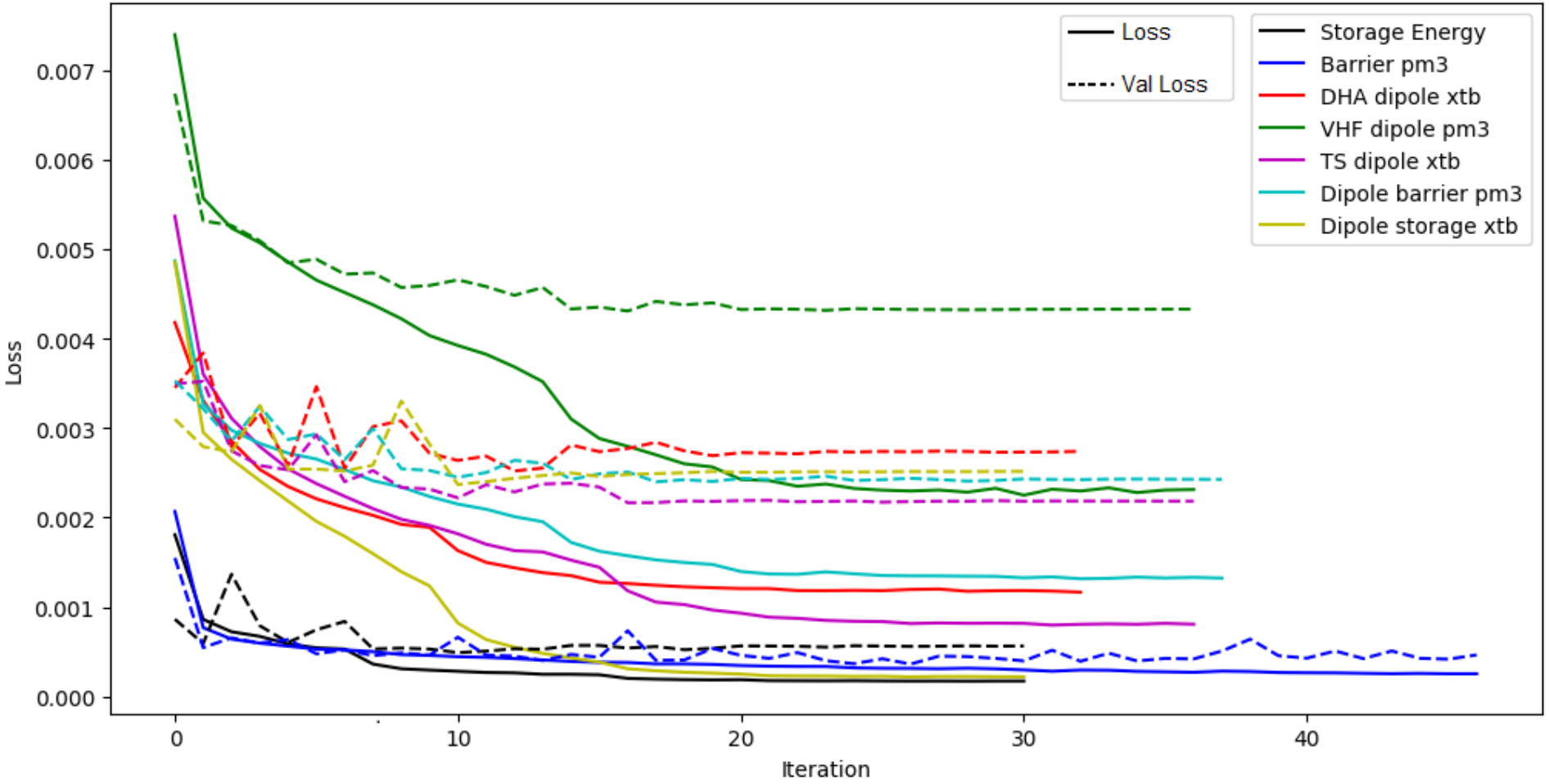


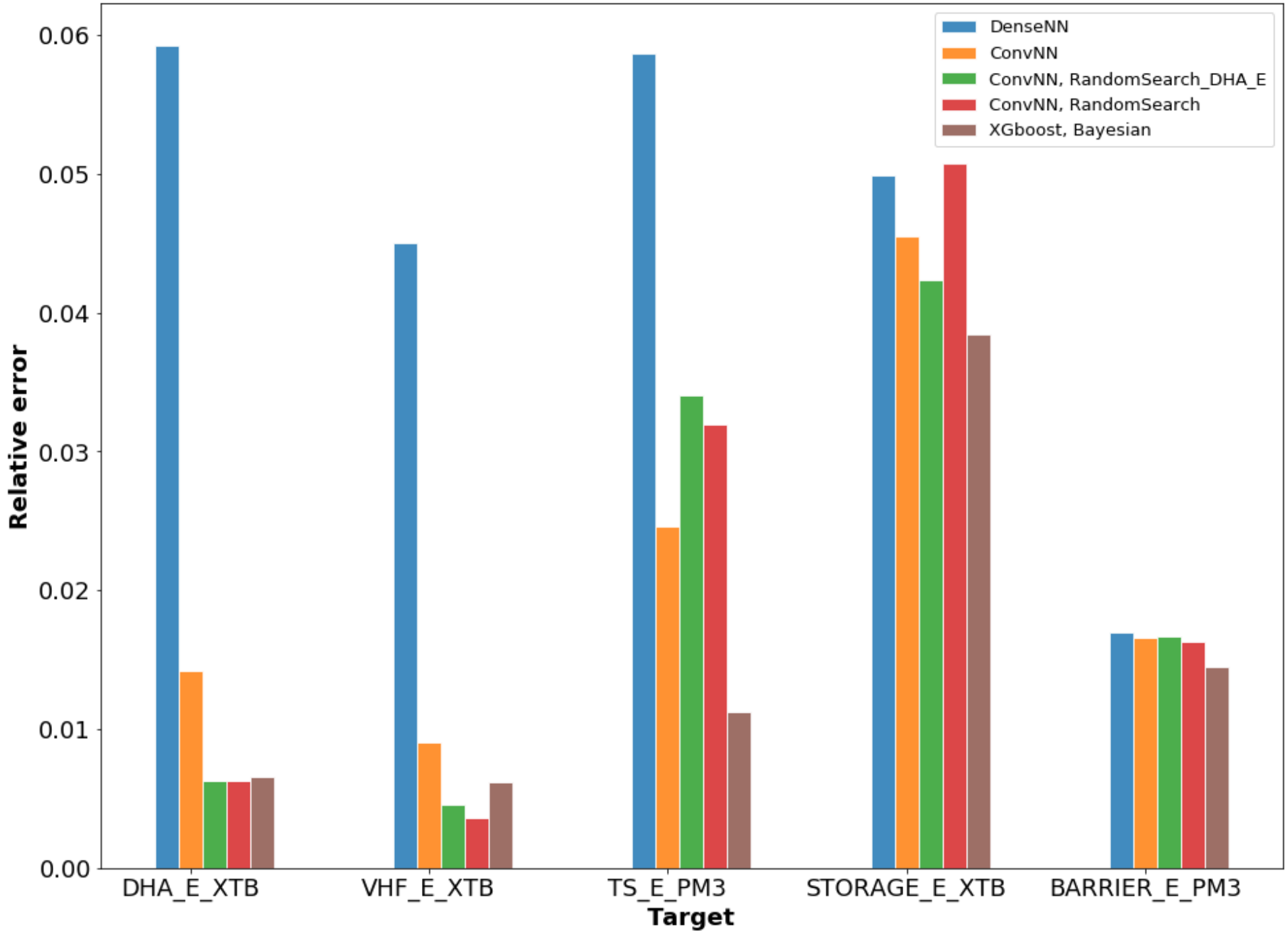

DenseNN
ConvNN
ConvNN, RandomSearch_DHA_E
ConvNN, RandomSearch
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Relative error
DHA
VHF
TS
STO.EN.
BAR
Target

Training vs. Validation Loss

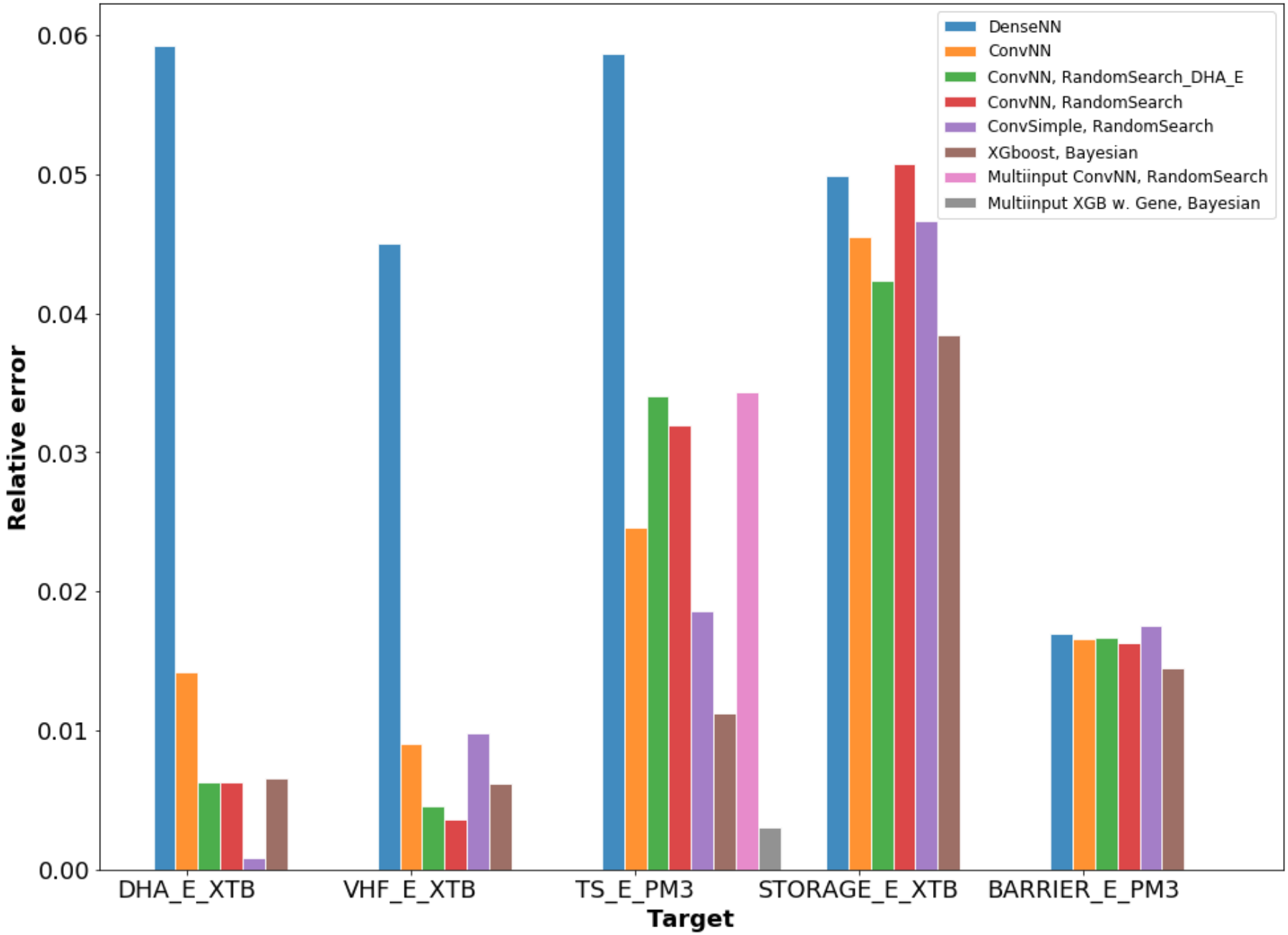


Training vs. Validation Loss

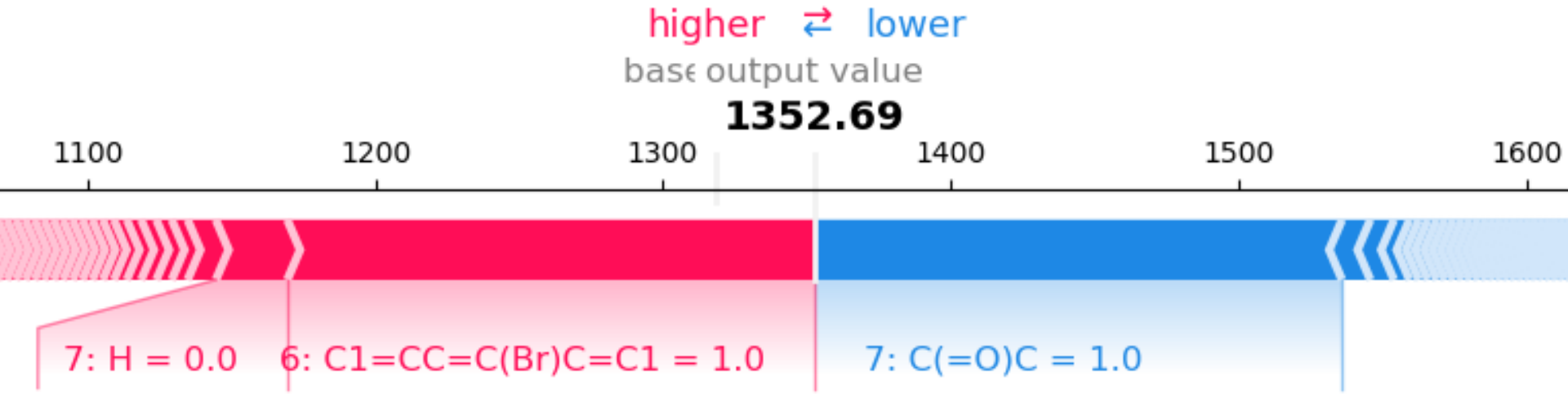


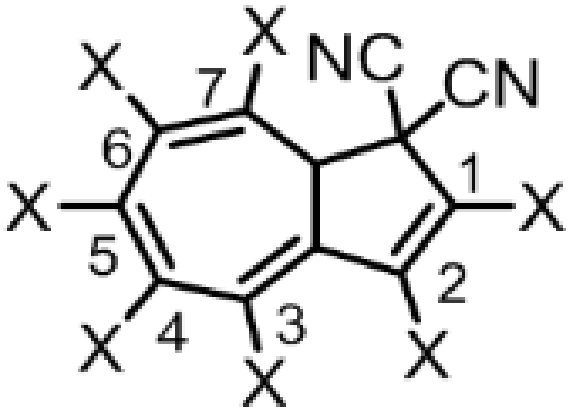

DenseNN
ConvNN
ConvNN, RandomSearch_DHA_E
ConvNN, RandomSearch
XGboost, Bayesian
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Relative error
DHA_E_XTB
VHF_E_XTB
TS_E_PM3
STORAGE_E_XTB
BARRIER_E_PM3
Target

Predicting TS energies from DHA/VHF features

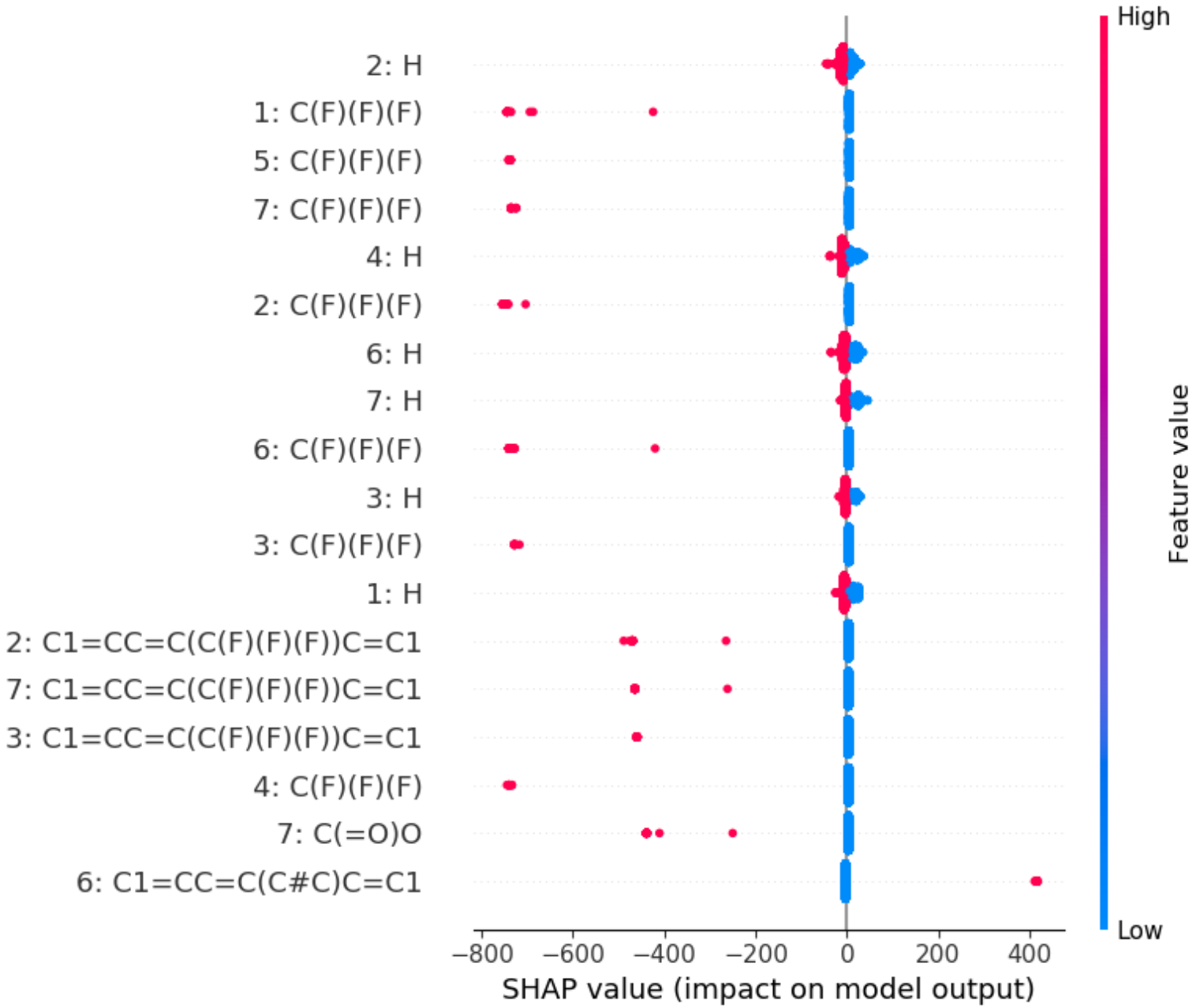


SHAP –which features are most important?





SHAP –which features are most important?



Summary

- Created own data
- Tested 3 different ML models
- Able to predict electrochemical properties from simple molecular representation
- Able to predict non-trivial TS energy from simpler to simulate properties
- Model much faster for evaluation than simulating/synthesizing individual molecules

Outlook

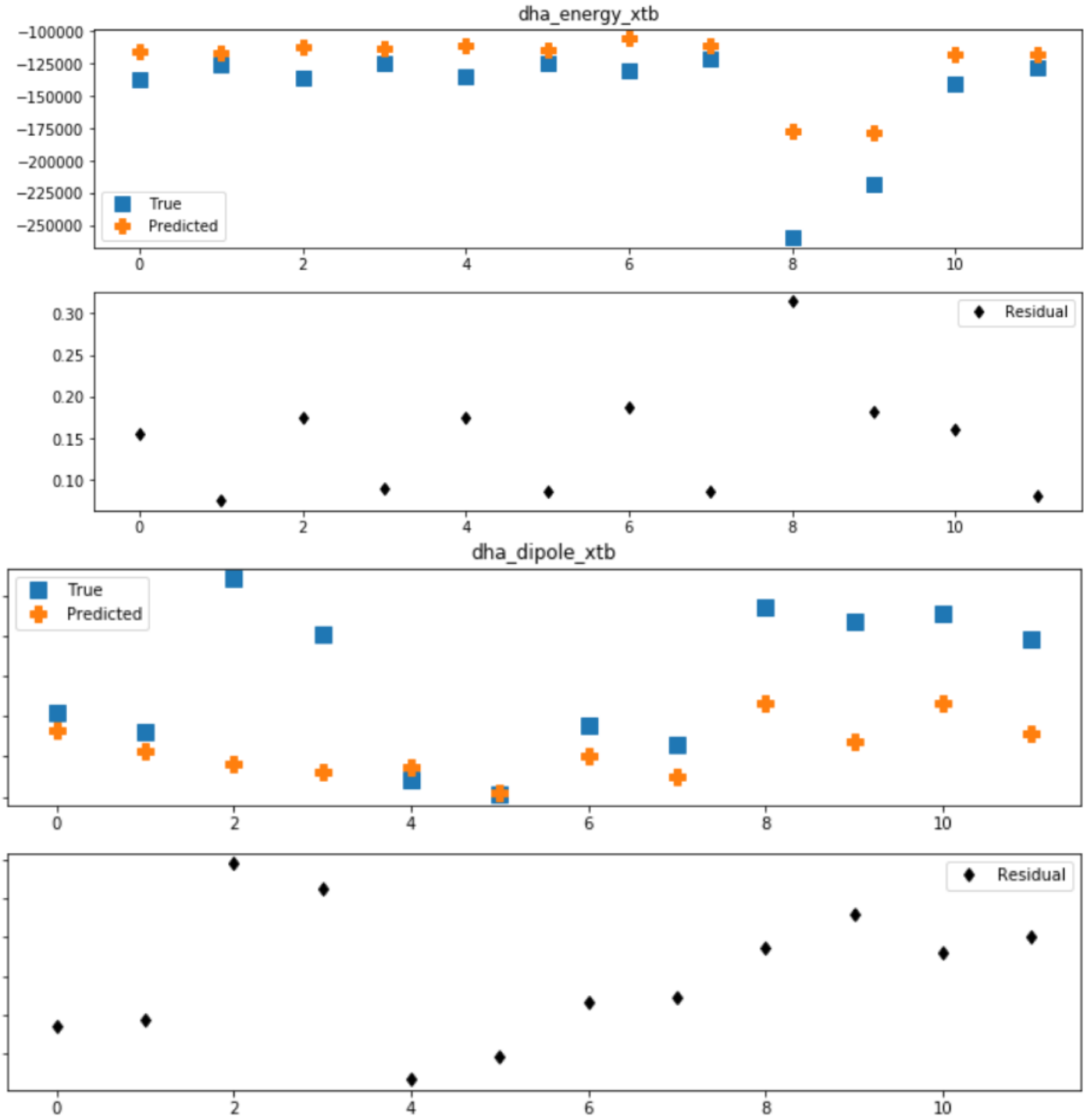
- Dense layers added little increased accuracy
- Hyperparameter search more thorough, but time constraints
- Expanding the database with triply, quadruply substituted VHF

Appendix

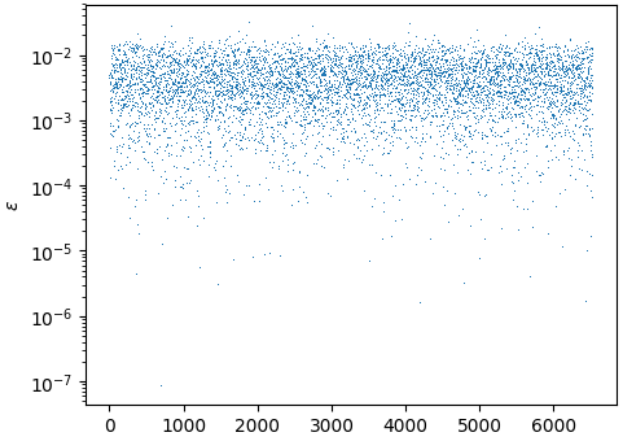
All group members have contributed evenly to the project

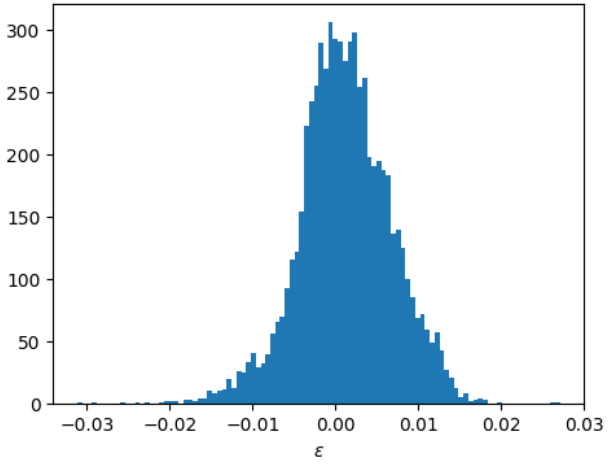
Testing for 3 and 4 substituents

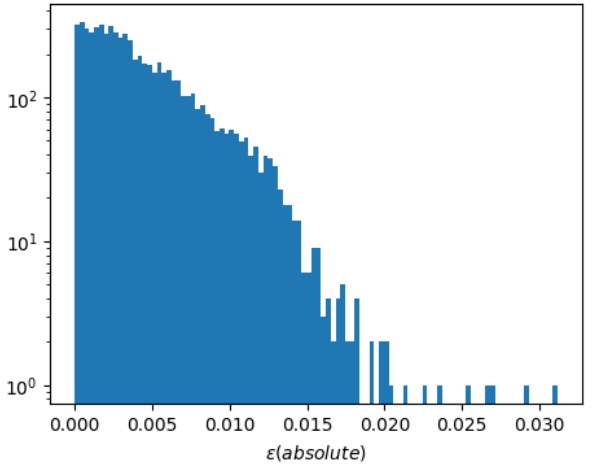
- Model worsens due to steric hindrance

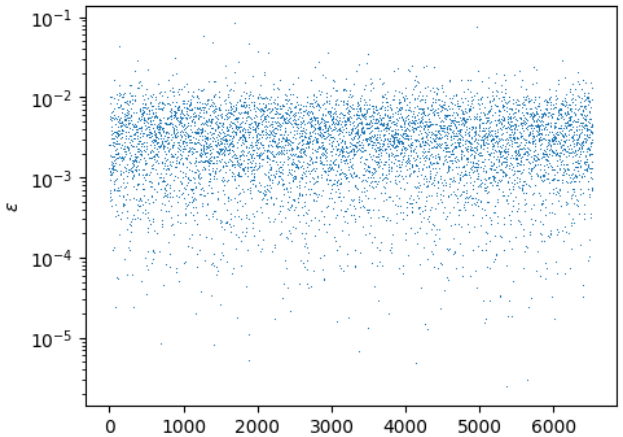


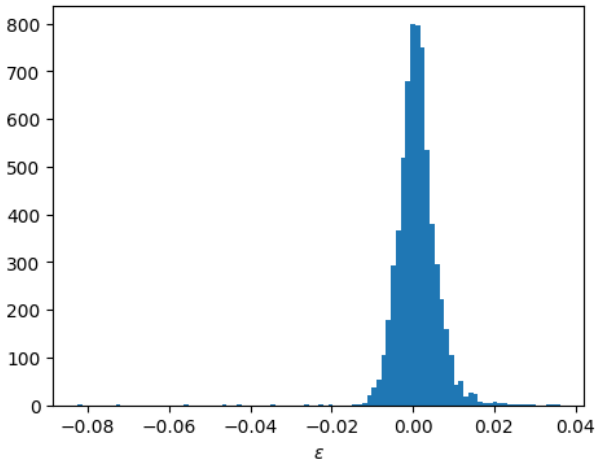
DHA and VHF Energy

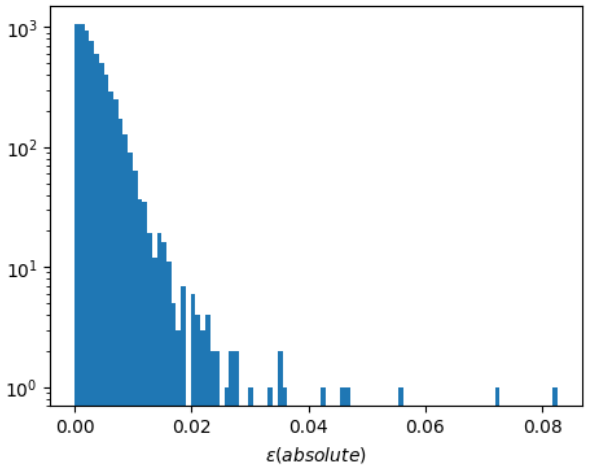




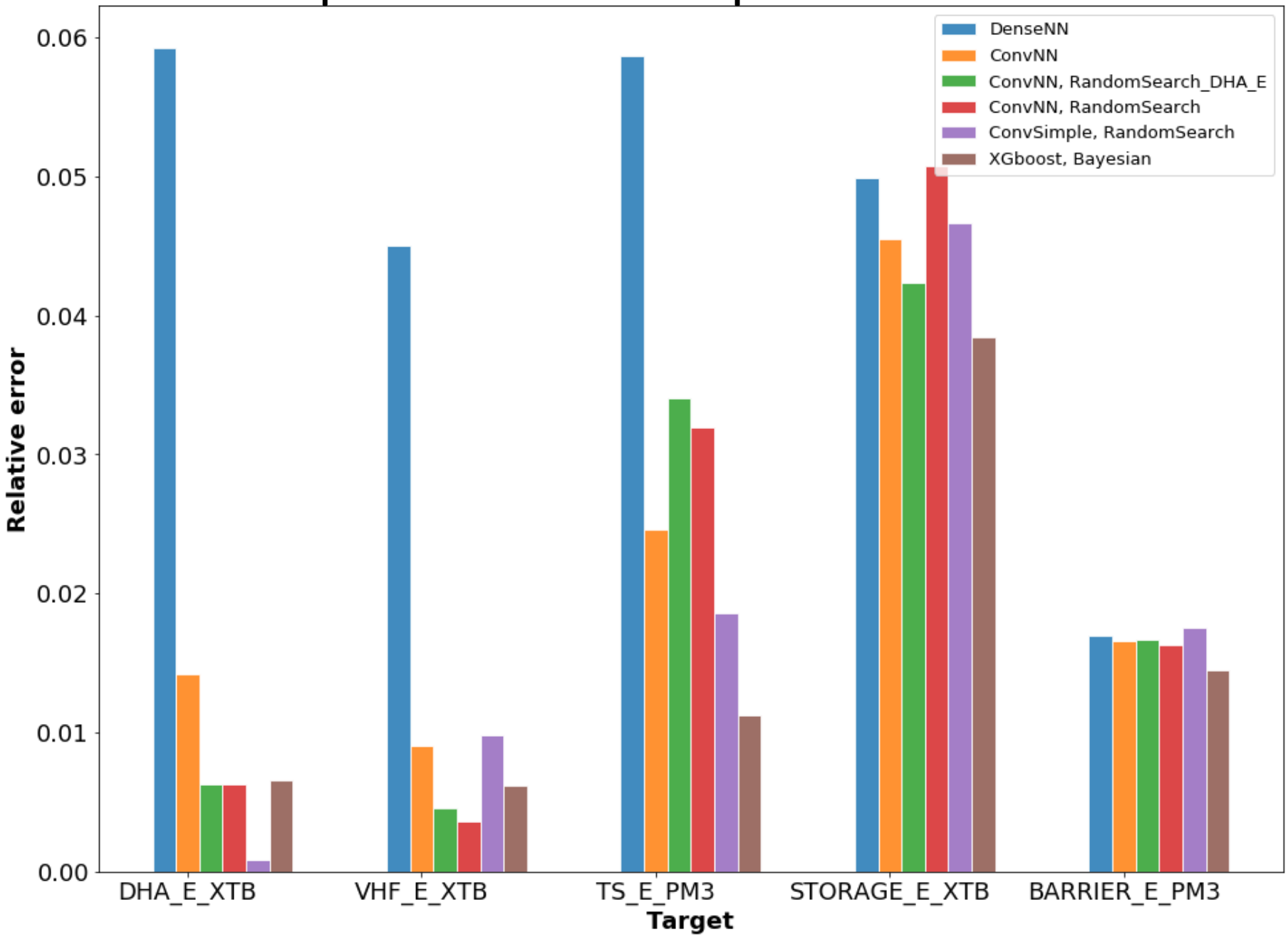








Comparison with Simple ConvNN

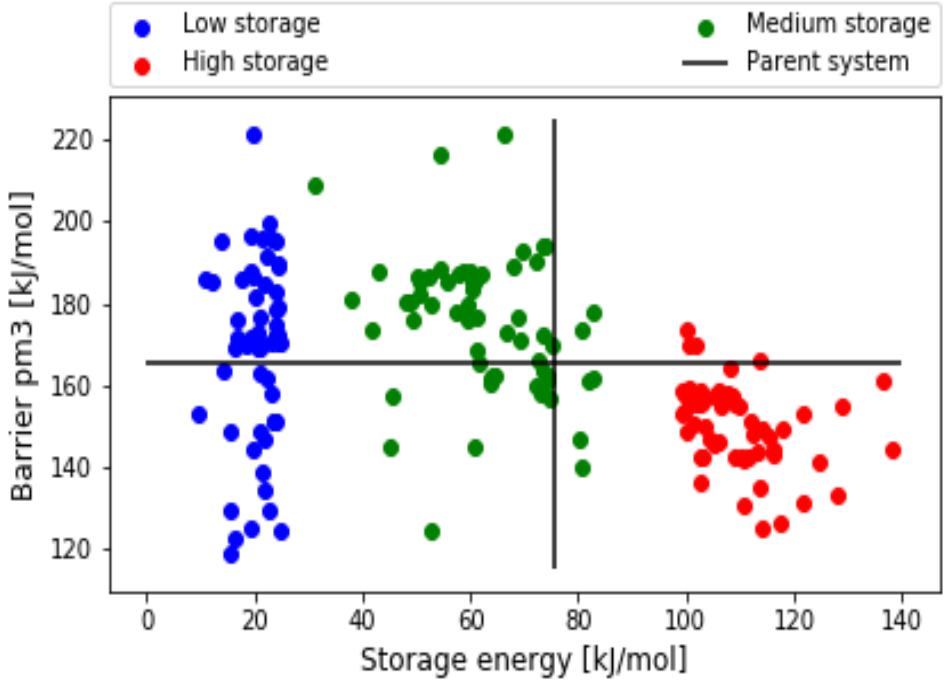


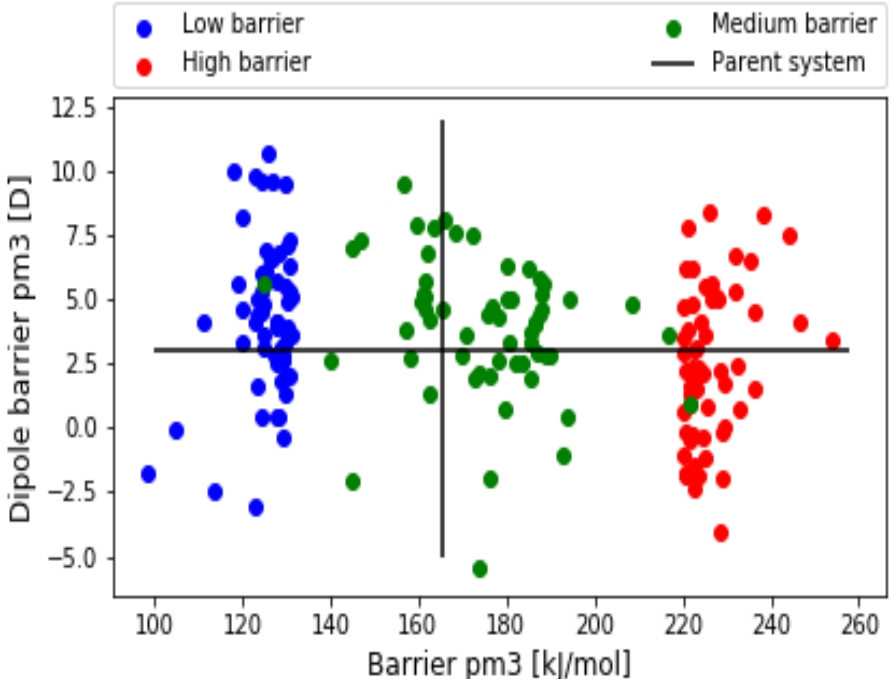
Full list of substituents

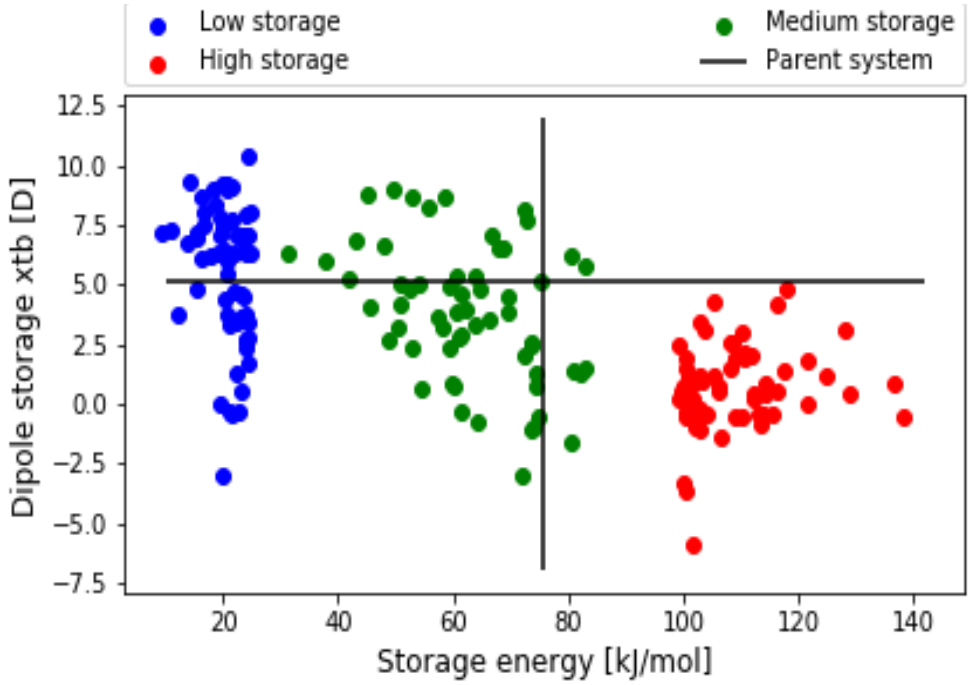
- ['H', 'F', 'Cl', 'Br', 'C(F)(F)(F)', 'C#N',
- '[N+](=O)([O-])', 'C(=O)[H]', 'C(=O)O',
- 'C(=O)C', 'C(=O)N', 'C#C', 'S(=O)(=O)(C)',
- 'C=N', 'O', 'OC', 'N', 'N(C)(C)', 'C',
- 'N(C(=O)(C))', 'SC', 'C1=CC=C(F)C=C1',
- 'C1=CC=C(Cl)C=C1', 'C1=CC=C(Br)C=C1',
- 'C1=CC=C(C(F)(F)(F))C=C1',
- 'C1=CC=C(C#N)C=C1', 'C1=CC=C([N+](=O)([O-]))C=C1',
- 'C1=CC=C(C(=O)[H])C=C1', 'C1=CC=C(C(=O)O)C=C1',
- 'C1=CC=C(C(=O)C)C=C1', 'C1=CC=C(C(=O)N)C=C1',
- 'C1=CC=C(C#C)C=C1', 'C1=CC=C(S(=O)(=O)(C))C=C1',
- 'C1=CC=C(C=N)C=C1', 'C1=CC=C(O)C=C1',
- 'C1=CC=C(OC)C=C1', 'C1=CC=C(N)C=C1',
- 'C1=CC=C(N(C)(C))C=C1', 'C1=CC=C(C)C=C1',
- 'C1=CC=C(N(C(=O)(C)))C=C1', 'C1=CC=C(SC)C=C1',
- 'C1=CC=CC=C1']

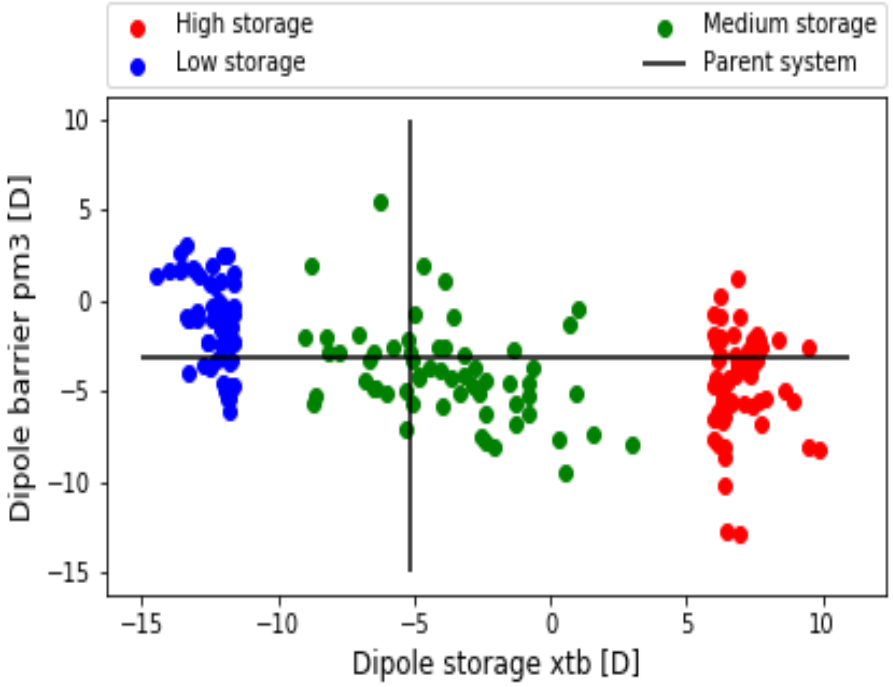
Full list of features

```
array(['name', 'gene', 'smiles', 'dha_energy_xtb', 'vhf_energy_xtb',
       'storage_energy', 'storage_density', 'ts_dipole_pm3',
       'vhf_dipole_pm3', 'dha_dipole_xtb', 'vhf_dipole_xtb',
       'ts_dipole_xtb', 'ts_energies_pm3', 'vhf_energies_pm3',
       'dha_dipole_x_pm3', 'ts_dipole_x_pm3', 'vhf_dipole_x_pm3',
       'dha_dipole_y_pm3', 'vhf_dipole_y_pm3', 'ts_dipole_y_pm3',
       'dha_dipole_z_pm3', 'ts_dipole_z_pm3', 'vhf_dipole_z_pm3',
       'dha_dipole_x_xtb', 'ts_dipole_x_xtb', 'vhf_dipole_x_xtb',
       'dha_dipole_y_xtb', 'vhf_dipole_y_xtb', 'ts_dipole_y_xtb',
       'dha_dipole_z_xtb', 'ts_dipole_z_xtb', 'vhf_dipole_z_xtb',
       'dha_qpole_xx_xtb', 'ts_qpole_xx_xtb', 'vhf_qpole_xx_xtb',
       'dha_qpole_yy_xtb', 'vhf_qpole_yy_xtb', 'ts_qpole_yy_xtb',
       'dha_qpole_zz_xtb', 'ts_qpole_zz_xtb', 'vhf_qpole_zz_xtb',
       'dha_qpole_xy_xtb', 'ts_qpole_xy_xtb', 'vhf_qpole_xy_xtb',
       'dha_qpole_xz_xtb', 'vhf_qpole_xz_xtb', 'ts_qpole_xz_xtb',
       'dha_qpole_yz_xtb', 'ts_qpole_yz_xtb', 'vhf_qpole_yz_xtb',
       'barrier_pm3', 'dipole_barrier_pm3', 'dipole_storage_xtb'],
      dtype=object)
```

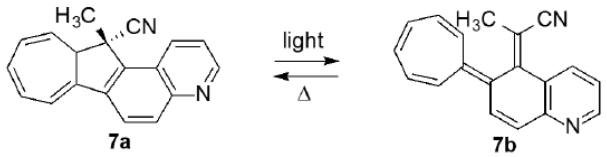

Low storage
High storage
Medium storage
Parent system
Barrier pm3 [kJ/mol]
Storage energy [kJ/mol]


Low barrier
High barrier
Medium barrier
Parent system
Dipole barrier pm3 [D]
Barrier pm3 [kJ/mol]


Low storage
High storage
Medium storage
Parent system
Dipole storage xtb [D]
Storage energy [kJ/mol]


High storage
Low storage
Medium storage
Parent system
Dipole barrier pm3 [D]
Dipole storage xtb [D]

Results



Units of kJ/mol / MJ/kg	vacuum	CH	CH_2Cl_2	MeCN
$\Delta H_{\mathbf{7a}\rightarrow\mathbf{7b}}$ MP2	140 / 0.52	140 / 0.52	141 / 0.52	140 / 0.52
$\Delta G_{\mathbf{TS}\leftarrow\mathbf{7b}}$ MP2	-	-	62.7	61.4
$\Delta G_{\mathbf{TS}\cdot\mathbf{H}^+\leftarrow\mathbf{7b}\cdot\mathbf{H}^+}$ MP2	65.8	-	59.2	59.9
$\Delta E_{\mathbf{7a}\rightarrow\mathbf{7b}}$ CCSD(T) SPE	133 / 0.49	133 / 0.49	133 / 0.49	133 / 0.49

Table 3 Energy storage capacities and thermal back-reaction barriers calculated at the MP2/6-311+G(d) level of theory and change in single point energy and the corresponding specific energy calculated at the CCSD(T)/6-311+G(d) level of theory on M06-2X/6-311+G(d) optimized structures of systems **7** and $\mathbf{7}\cdot\mathbf{H}^+$ *in vacuo* and solvents.

Half-life shortened by a factor of 4

Half-life shortened by a factor of 2

$$t_{1/2} \propto \frac{1}{k_r} \propto \exp\left(\frac{\Delta G_{\mathrm{TS}\leftarrow\mathrm{VHF}}}{k_B T}\right)$$