Ai Paper and Project

PAPER

[graduated thesis] :
Advanced model prediction by
hyperparameter optimization in chemistry

PROJECT

- ❖ **3D Camera project**
- ❖ **Mass-Spectrum Prediction**

Seung Jin Lee

PAPER

[graduated thesis] :

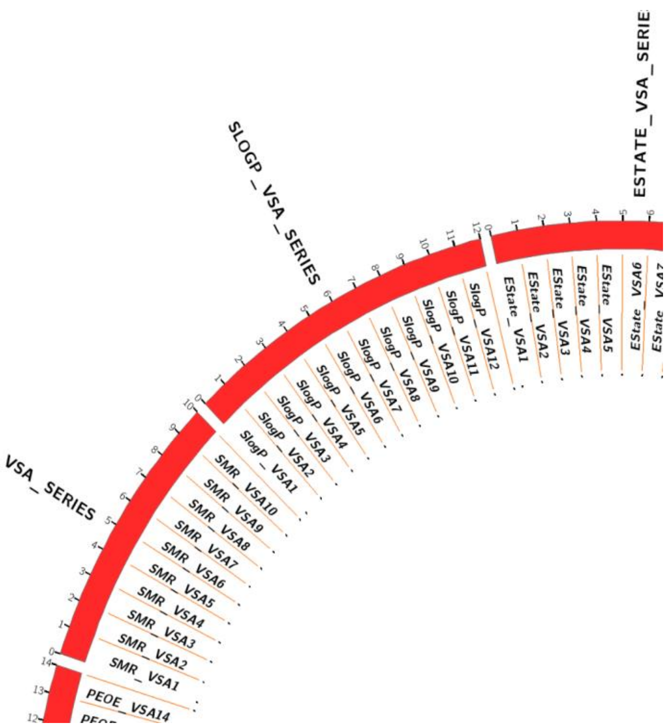
Advanced model prediction by hyperparameter optimization in chemistry

(Lab2. 2020.10 – 2022.02)

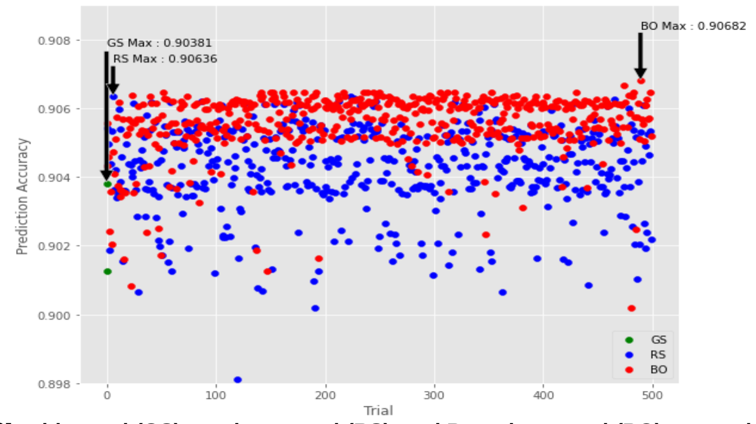
Summary: We achieved a significant improvement in predicting solubility, which is used in drug development, protein docking, pesticides, and toxins, compared to existing methodologies, with an R2 score increase of 0.2-0.3. Our highest score was 0.991.

methodology (AFO) employs two approaches to calculate or explore maximizing performance by reflecting the typical 49 chemical descriptors (features) used in analytical chemistry in the input data to predict solubility, which is used in drug development, protein docking, pesticides, and toxins, etc. The methodology uses Bayesian Optimization of the Search method but utilizes the surrogate model of TPE (Tree-Parzen Estimator) to achieve higher processing speed and performance than Gaussian. In the second method, hyperparameter optimization of the model using the same TPE-Bayesian Optimization is attempted to maximize performance.

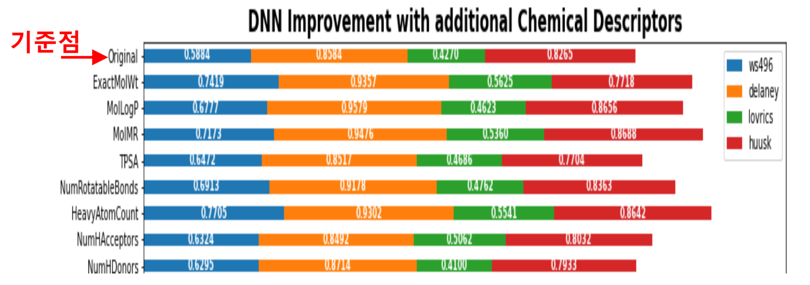
language : Python, C++, CUDA
Library : tensorflow, pytorch, scikit-learn, sqlite, postgresql, RDKit, optuna



[1] **List of 49 chemical descriptors**

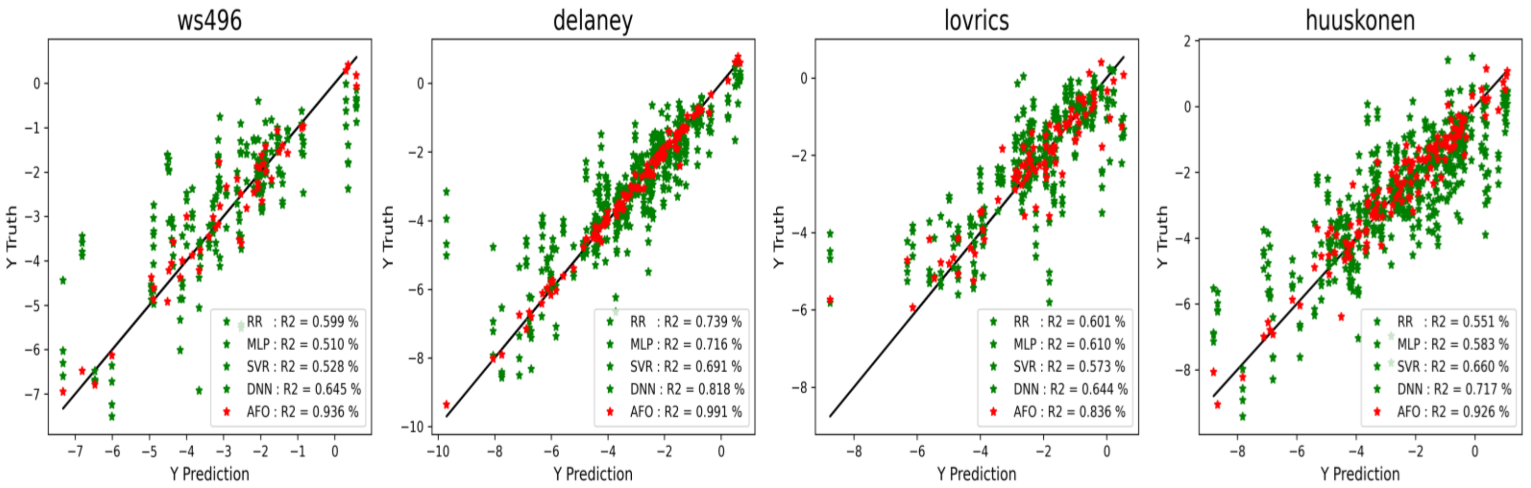


[2] grid search(GS), random search(RS), and Bayesian search(BO) comparison



	ws496	delaney	lovrics	huusk
Original	0.5884	0.8584	0.4270	0.8265
ExactMolWt	0.7419	0.9357	0.5625	0.7718
MolLogP	0.6777	0.9579	0.4623	0.8656
MolMR	0.7173	0.9476	0.5360	0.8688
TPSA	0.6472	0.8517	0.4686	0.7704
NumRotatableBonds	0.6913	0.9178	0.4762	0.8363
HeavyAtomCount	0.7705	0.9302	0.5541	0.8642
NumHAcceptors	0.6324	0.8492	0.5062	0.8032
NumHDonors	0.6295	0.8714	0.4100	0.7933

[3] Results of individual chemical descriptors reflected in input data



	ws496	delaney	lovrics	huuskonen
RR	R2 = 0.599 %	R2 = 0.739 %	R2 = 0.601 %	R2 = 0.551 %
MLP	R2 = 0.510 %	R2 = 0.716 %	R2 = 0.610 %	R2 = 0.583 %
SVR	R2 = 0.528 %	R2 = 0.691 %	R2 = 0.573 %	R2 = 0.660 %
DNN	R2 = 0.645 %	R2 = 0.818 %	R2 = 0.644 %	R2 = 0.717 %
AFO	R2 = 0.936 %	R2 = 0.991 %	R2 = 0.836 %	R2 = 0.926 %

[4] Predicted results for different datasets. It can be seen that AFO (our method) obtains the highest score in all datasets

PROJECT

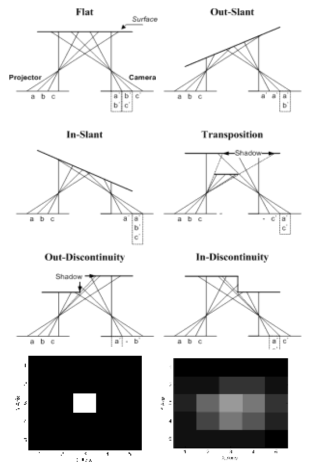
❖ 3D Camera project
(Lab1. 2018.10 – 2019.06)

Summary: Responsible for improving the accuracy and computation speed of 3D cameras, the error calibration was improved from 0.097 to **0.081** and the computation time was reduced from 37 seconds to **24 seconds**.

Methodology: The 3D camera technique involves a shooting method using two cameras with overlapping areas. It is necessary to minimize the **error calibration** that occurs in the overlapping areas. To improve this, the unnecessary loop statement and register setting of the **Hierarchical Orthogonal codec** were simplified, and the error rate was reduced by assigning forced fixed values to OpenCV and OpenGL. Additionally, Nvidia GPU CUDA was utilized to further reduce computation time.

language : Python, C++, CUDA
Library : OpenCV, OpenGL, Camera SW

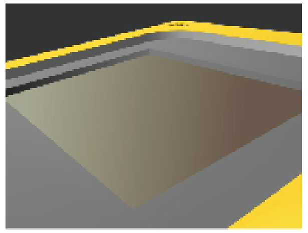
workspace



Surface geometries & signal mixturing

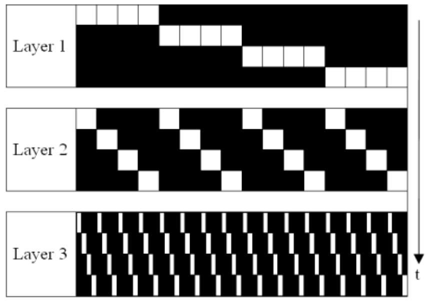
Surface reflectance

Single Mixing

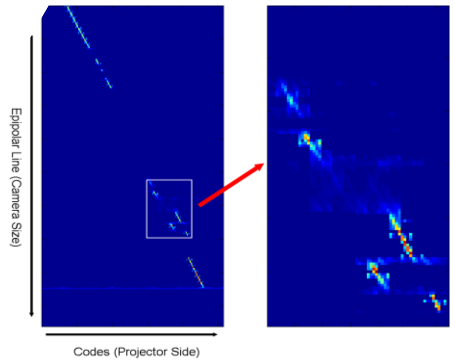

(Projector)

Captured Images (Camera)

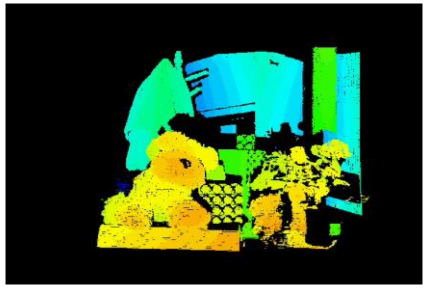
CODEC algorithm



Hierarchical Orthogonal Coding



Signal Separation Decoding


3D Point Cloud

PROJECT

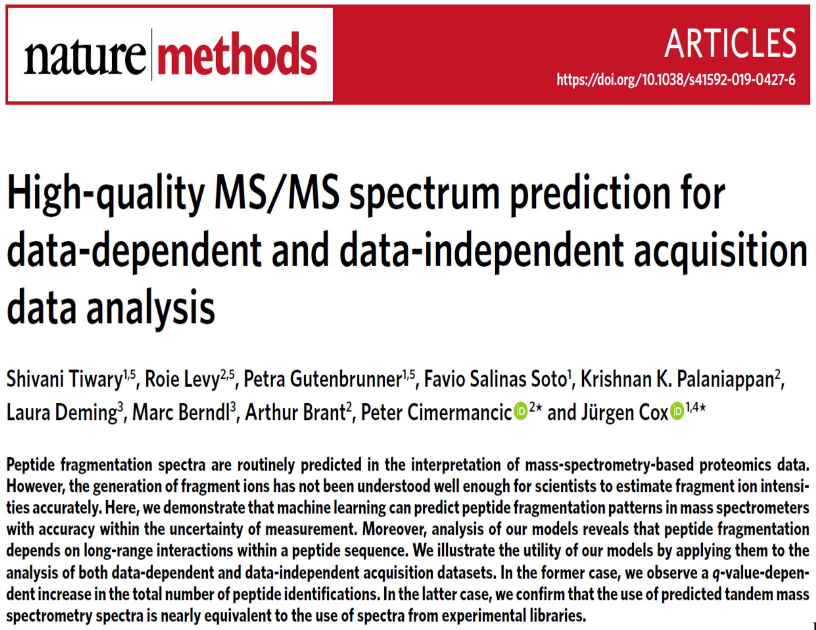
❖ Mass-Spectrum Prediction

(2022.05 – 2022.06)

Summary: A model was developed to predict Mass Spectrum (MS) data, achieving a higher score of 0.998 than the evaluation index of the latest paper, PCC 0.957.

Method: In the given data x, additional metadata is included based on the given protein sequence for training, and y values are matched to the intensity of molecules based on the given x values for training.

The top paper's model used the Bi-LSTM encoding method, which was useful for sequence learning in large-scale data. However, here, the latest techniques such as Attention algorithm and hyperparameter optimization were added to increase the potential of the existing model while improving the quality of the data. Thus, feature analysis was attempted based on such work. As a result of these attempts, a higher score of PCC 0.998 was obtained than the previous score.

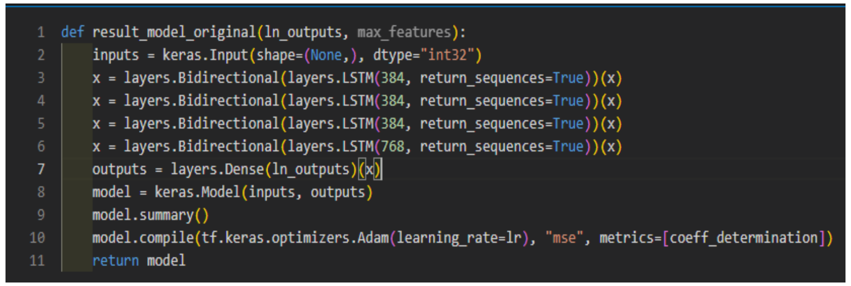

nature methods | ARTICLES

https://doi.org/10.1038/s41592-019-0427-6

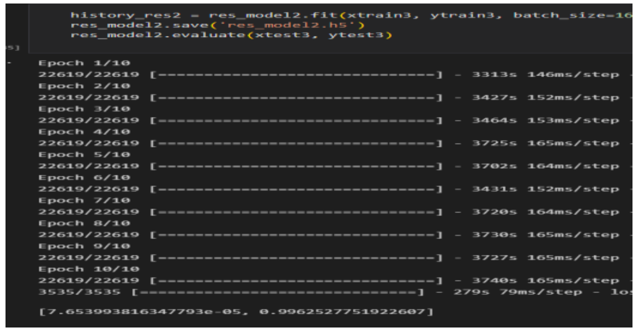
High-quality MS/MS spectrum prediction for data-dependent and data-independent acquisition data analysis

Shivani Tiwary[1,5], Roie Levy[2,5], Petra Gutenbrunner[1,5], Favio Salinas Soto[1], Krishnan K. Palaniappan[2], Laura Deming[3], Marc Berndl[3], Arthur Brant[2], Peter Cimermancic[2*] and Jürgen Cox[1,4*]

Peptide fragmentation spectra are routinely predicted in the interpretation of mass-spectrometry-based proteomics data. However, the generation of fragment ions has not been understood well enough for scientists to estimate fragment ion intensities accurately. Here, we demonstrate that machine learning can predict peptide fragmentation patterns in mass spectrometers with accuracy within the uncertainty of measurement. Moreover, analysis of our models reveals that peptide fragmentation depends on long-range interactions within a peptide sequence. We illustrate the utility of our models by applying them to the analysis of both data-dependent and data-independent acquisition datasets. In the former case, we observe a q-value-dependent increase in the total number of peptide identifications. In the latter case, we confirm that the use of predicted tandem mass spectrometry spectra is nearly equivalent to the use of spectra from experimental libraries.

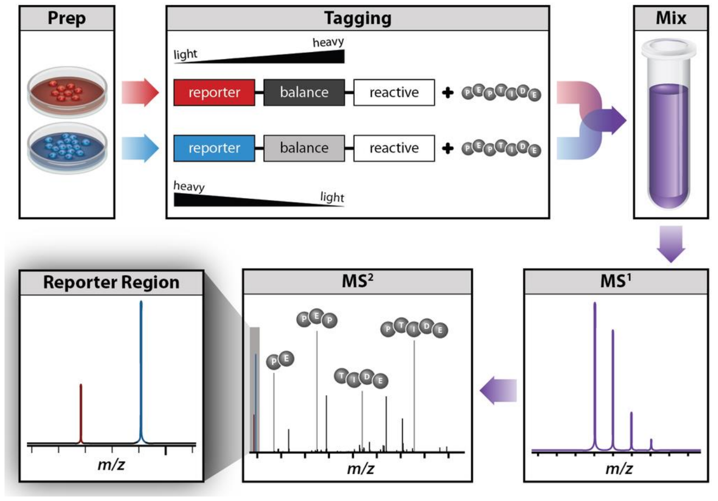

```python
def result_model_original(ln_outputs, max_features):
    inputs = keras.Input(shape=(None,), dtype="int32")
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(768, return_sequences=True))(x)
    outputs = layers.Dense(ln_outputs)(x)
    model = keras.Model(inputs, outputs)
    model.summary()
    model.compile(tf.keras.optimizers.Adam(learning_rate=lr), "mse", metrics=[coeff_determination])
    return model
```

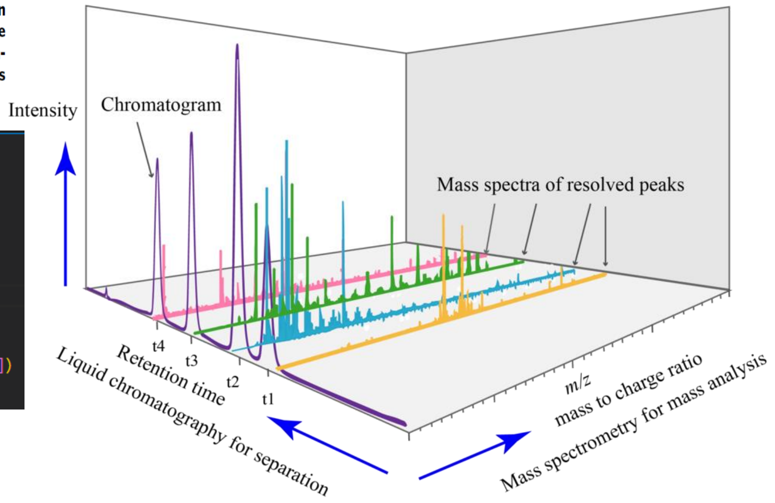
[1] target top-tier paper code representation


```
    history_res2 = res_model2.fit(xtrain3, ytrain3, batch_size=16
    res_model2.save('res_model2.h5')
    res_model2.evaluate(xtest3, ytest3)

Epoch 1/10
22619/22619 [==============================] - 3313s 146ms/step
Epoch 2/10
22619/22619 [==============================] - 3427s 152ms/step
Epoch 3/10
22619/22619 [==============================] - 3464s 153ms/step
Epoch 4/10
22619/22619 [==============================] - 3725s 165ms/step
Epoch 5/10
22619/22619 [==============================] - 3702s 164ms/step
Epoch 6/10
22619/22619 [==============================] - 3431s 152ms/step
Epoch 7/10
22619/22619 [==============================] - 3720s 164ms/step
Epoch 8/10
22619/22619 [==============================] - 3730s 165ms/step
Epoch 9/10
22619/22619 [==============================] - 3727s 165ms/step
Epoch 10/10
22619/22619 [==============================] - 3740s 165ms/step
3535/3535 [==============================] - 279s 79ms/step - lo

[7.653993816347793e-05, 0.9962527751922607]
```

R2 score evaluation (with test data) : **0.99625, (PCC 0.998)**

[3] optimized the preprocessing dataset and applied new algorithms




Intensity
Chromatogram
Mass spectra of resolved peaks
t4
t3
t2
t1
Retention time
Liquid chromatography for separation
m/z
mass to charge ratio
Mass spectrometry for mass analysis

Given data :

x = (**modified sequences**, retention time, mass to charge ratio)

y = (relative**) Intensity**

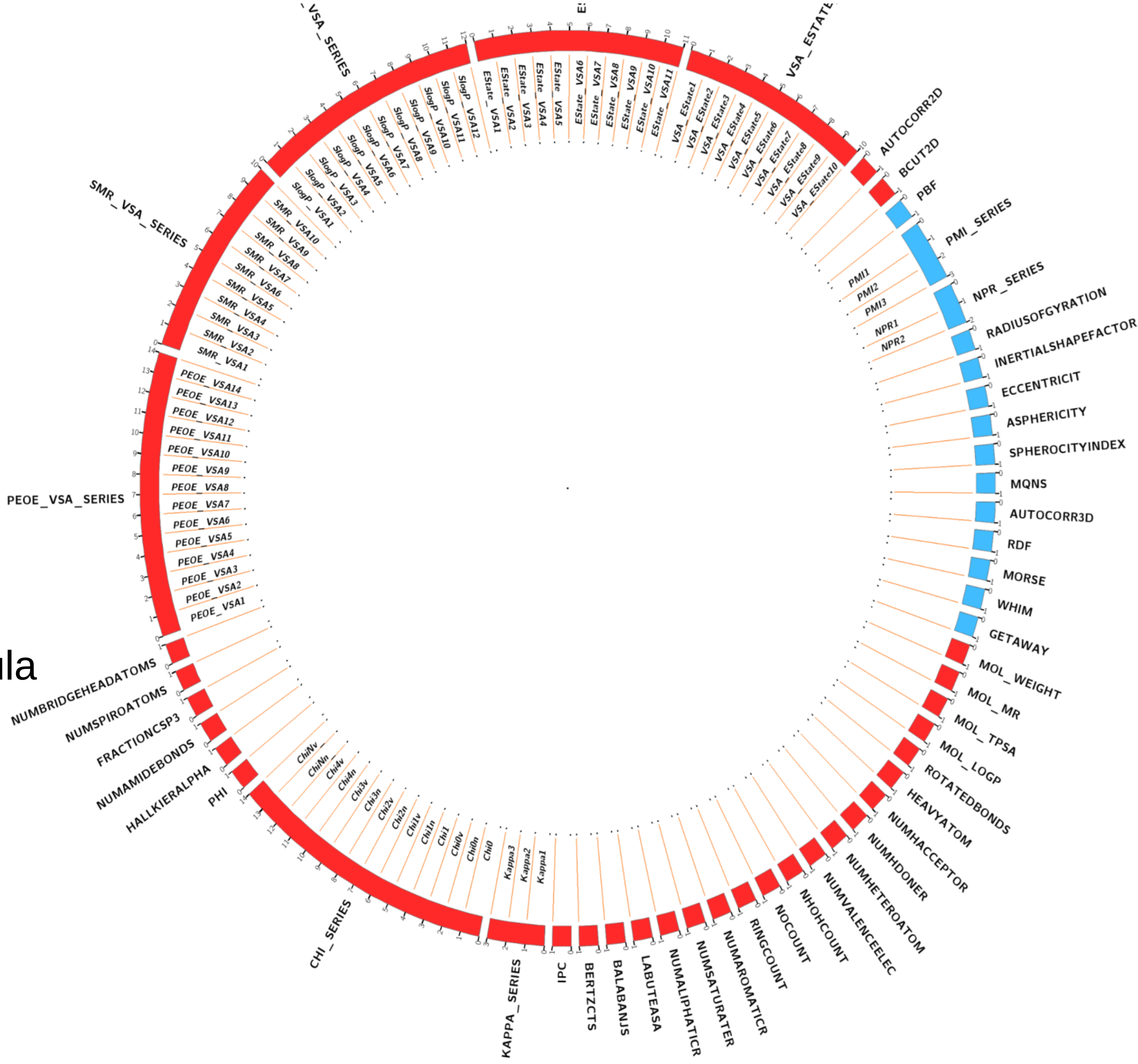
[2] The process of the MS experiment and explanation of data about x and y input setup

[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ Chemical Descriptors

- There are **49** chemical descriptors generating by open-sources.
- **2D** descriptors **(32)** – **Red** color
- **3D** descriptors **(14)** – **Blue** color

- Calculated from simple summation to complex calculation to complex formula by combination of geometry, graph-theory, derivation matrix, polarity, and so on.



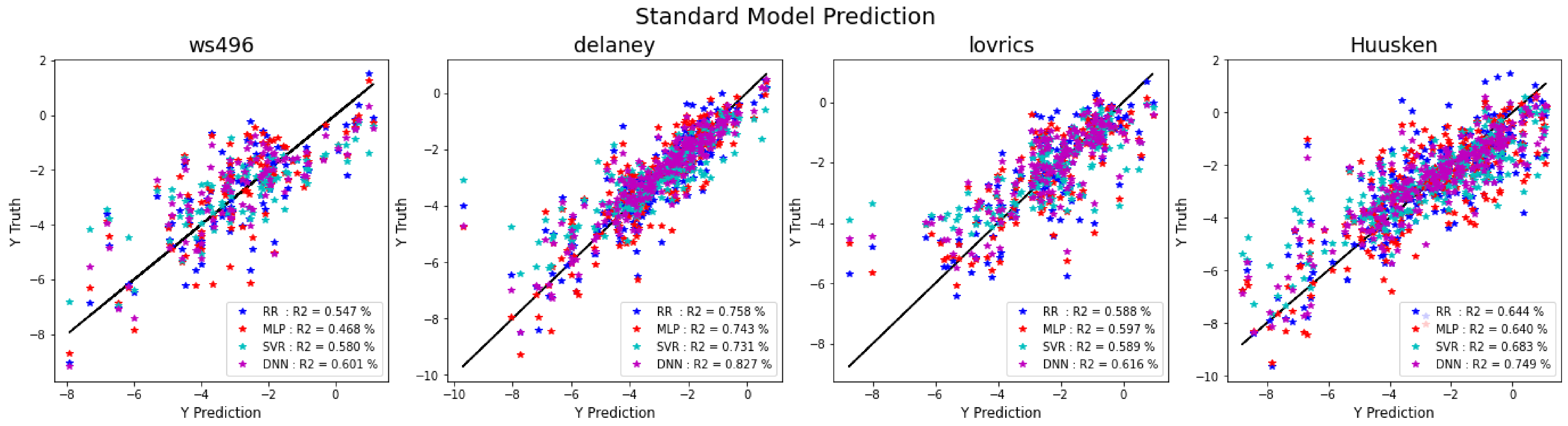
[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ Comparing standard models

- Before applying the our new method, it is necessary to analyze the standard model performance.
- In chemistry research, following algorithms are the popular to use the specific target data such as Ridge regression (RR), Multilayer Perceptron (MLP) and Support Vector Regression (SVR)and Deep Neural Network (DNN).

▪ RESULT:

Following the implementation of Deep Neural Network (DNN) is much higher score than Ridge regression (RR), Multilayer Perceptron (MLP) and Support Vector Regression (SVR) in R2 score.



[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ Our method (Automatic Feature Optimization, AFO)

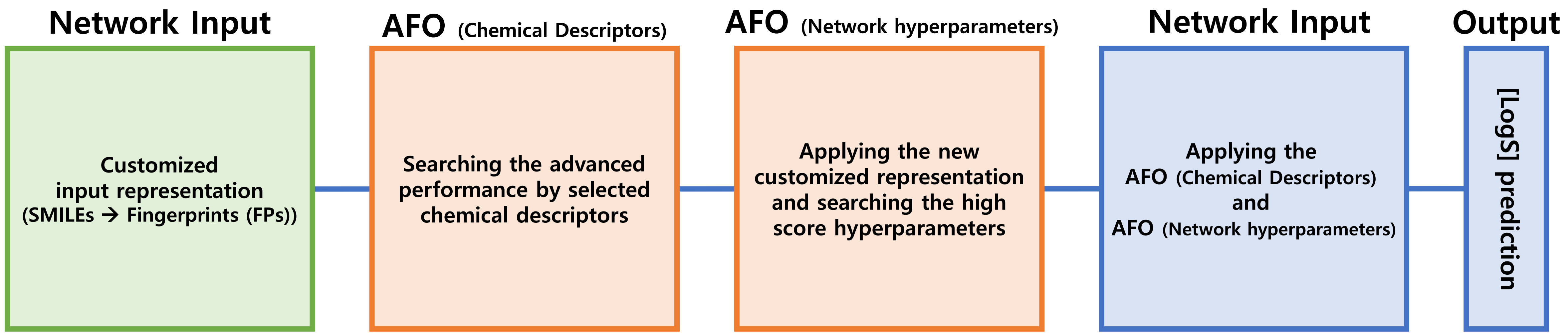
- Based-on the DNN model architecture, it is important to manage the hyperparameters.
- However, it is necessary to check out the improvement after customizing the input representation
- To solve above suggestion, implementing the search algorithm with TPE method for finding out the suitable chemical descriptors to given dataset.
- After improvement of input, it can be helpful to increase the score such as r2 score by hyperparameter search.

Algorithm : pseudocode AFO (using the Bayesian Optimization based on TPE)

```
Given: given input data $\mathbf{D_{i=1:n}} = \{\mathbf{x_i}, \mathbf{y_i}\}$ with SMILEs $\mathbf{x_i}$ and target logS $\mathbf{y_i}$.
The simple DNN model $\mathbf{f}$, surrogate model $\mathbf{f}^*$, acquisition function $\boldsymbol{\mathcal{L}}$ and hyperparameter $\boldsymbol{\xi}$
Goal: $\xi \in arg\max_{\xi}\{f^*(\xi)\}$
Initialize:
    $\hat{\mathbf{x}}_\mathbf{i} \leftarrow$ customized fingerprint
    $\widehat{\mathbf{D}}_{\mathbf{i=1:n}} = (\hat{\mathbf{x}}_\mathbf{i}, \mathbf{y_i})$
    $\mathbf{t}$ = HBO trial count
for $\mathbf{j = 1{:}t}$ do:
    Select $\xi$ of $\mathcal{L}$ by using HBO and initialize the $f^*$
    for $\mathbf{i = 1{:}n}$ do:
        Update $f^*$ by $\widehat{D}_i$ managing by given $\xi$
        Score $S_j$ by $f^*(\hat{x}_i|\xi)$
    End for
    If $\mathbf{S_j} \geq \mathbf{S_{j-1}}$:
        Update $\xi$
    endif
End for
return: the $\xi$ evaluated highest score
```

[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ Our method (Automatic Feature Optimization, AFO)


Network Input
Customized input representation (SMILEs → Fingerprints (FPs))
AFO (Chemical Descriptors)
Searching the advanced performance by selected chemical descriptors
AFO (Network hyperparameters)
Applying the new customized representation and searching the high score hyperparameters
Network Input
Applying the AFO (Chemical Descriptors) and AFO (Network hyperparameters)
Output
[LogS] prediction

- Using the Morgan / MACCS / Avalon FPs
- The length of FPs 2727

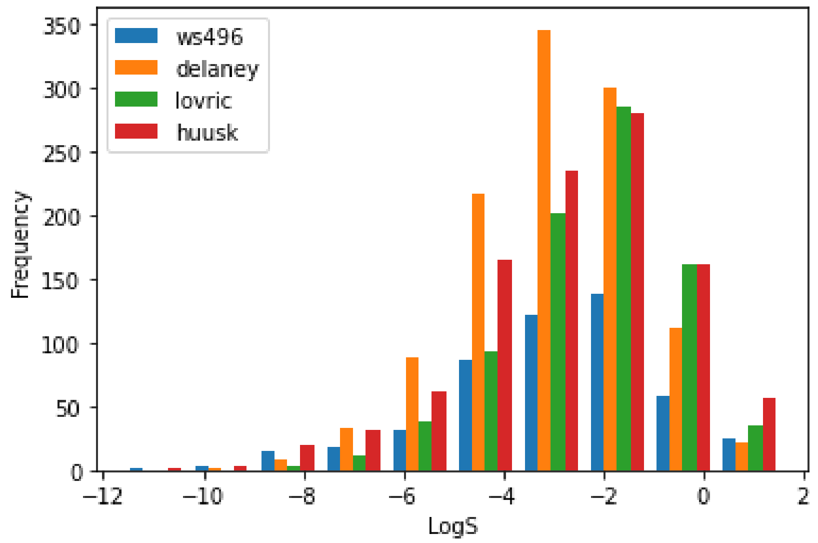
- The Chemical 49 descriptors
- Described as 2D and 3D representation to the descriptors

- Searching the five hyperparameters such as learning rate, dropout rate, regularization rate, the number of layers in mode, and nodes in each layers.

[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ Dataset

[1] **ws496** (2006, J. Chem. Inf. Model. 2006, 46, 2, 487–494)
[2] **Delaney** (2004, J. Chem. Inf. Comput. Sci. 2004, 44, 3, 1000–1005)
[3] **Lovrics** (M. Lovric, 829 drug-like molecules intrinsic solubility dataset." 2020)
[4] **huuskonen** (J. Chem. Inf. Comput. Sci. 2000, 40, 3, 773–777)

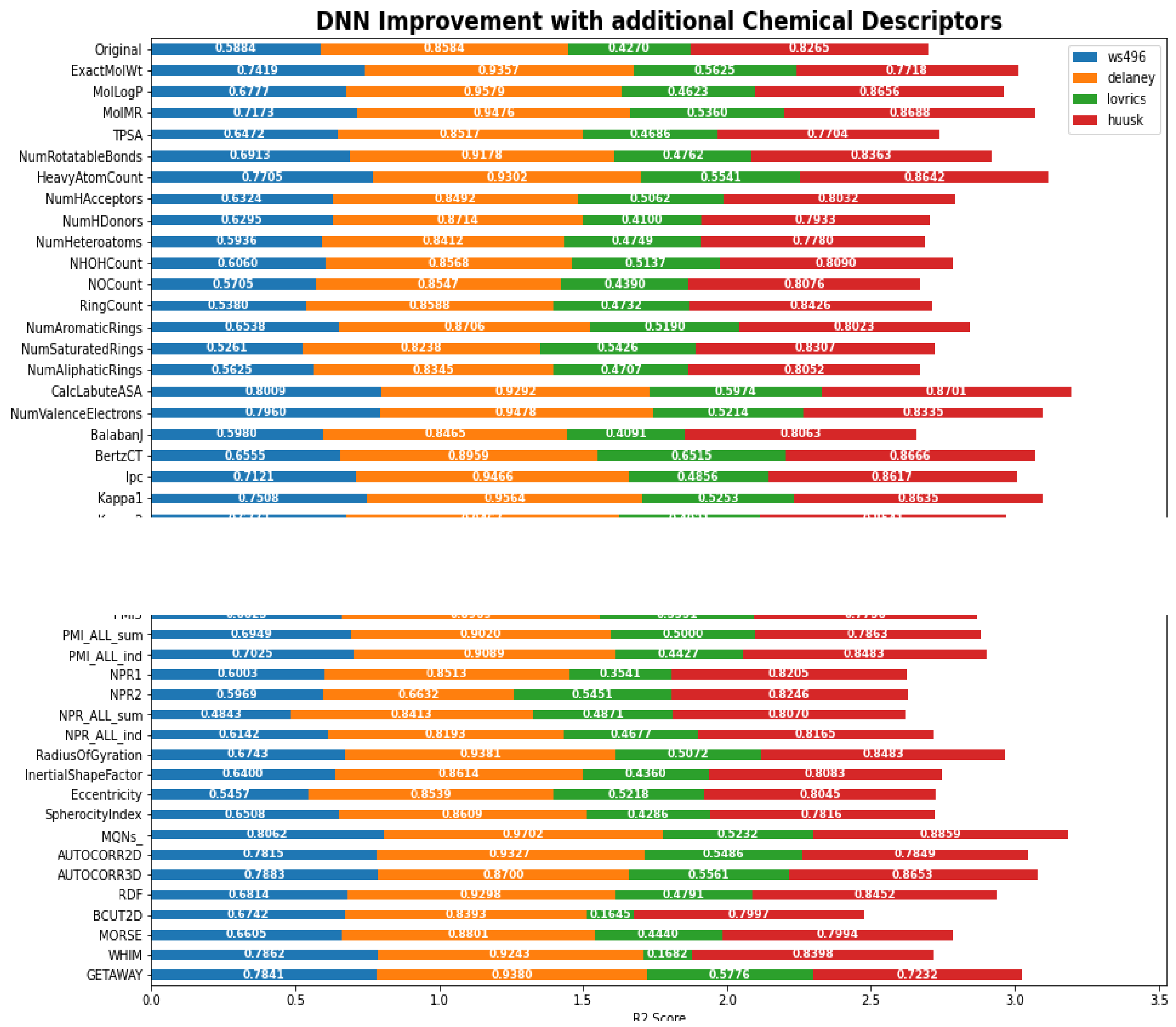


- Histogram of Solubility dataset: LogS for the ground truth value and Frequency for measuring how many times values appear.
- The range between -1.5 to -5.- is very high frequency of solubility molecules.
- The max value is under 2 and min value is over -12.0.

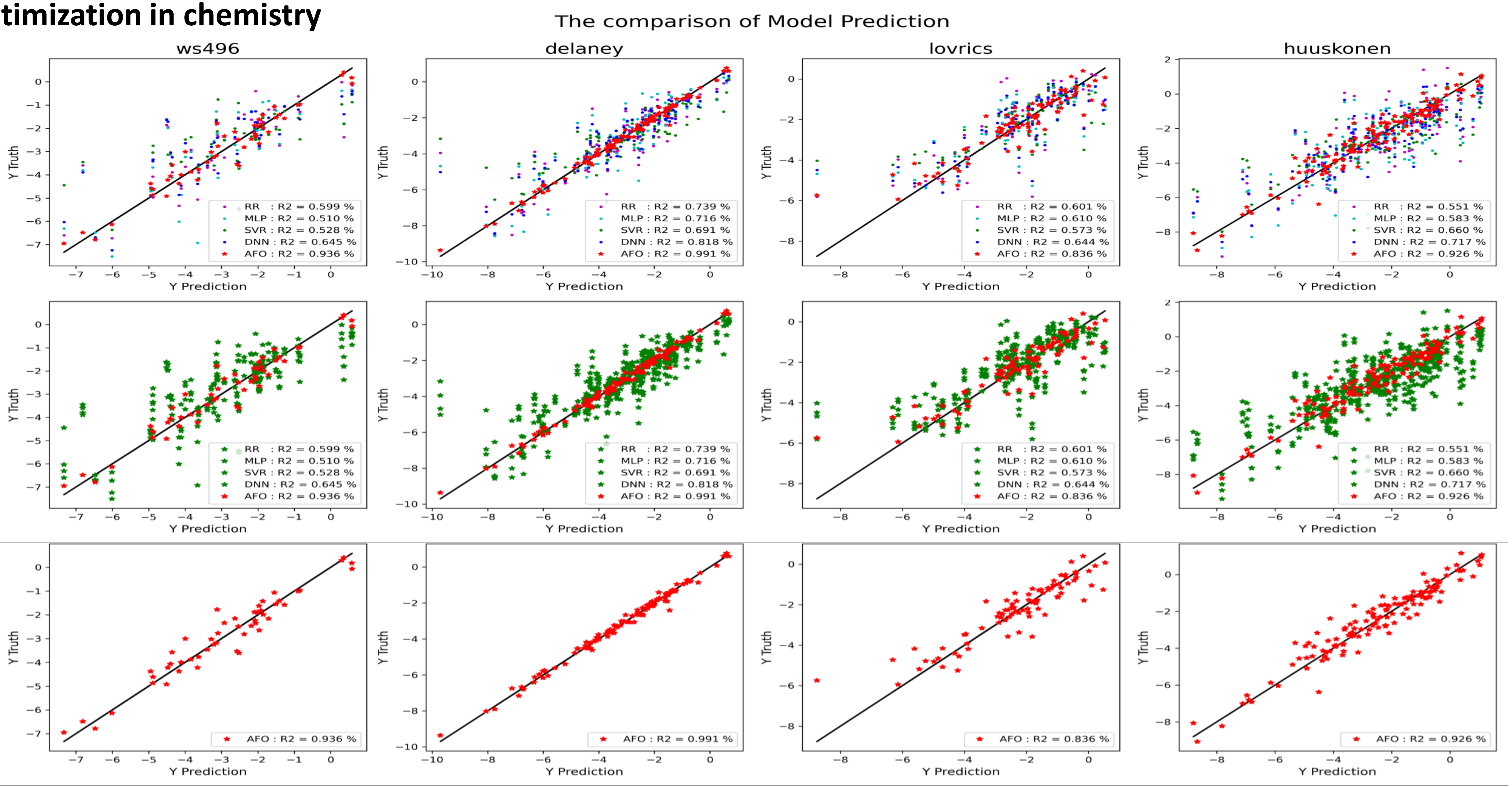
[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

❖ **Check the improvement by individual chemical descriptors.**

- we found that the unique values from chemical descriptors provided the high scores than ordinary and similar values.
- Depending on the information such as focused on the energy solubility, drug-solved, and protein-based solubility dataset, some descriptors helped to boost up the prediction performances.



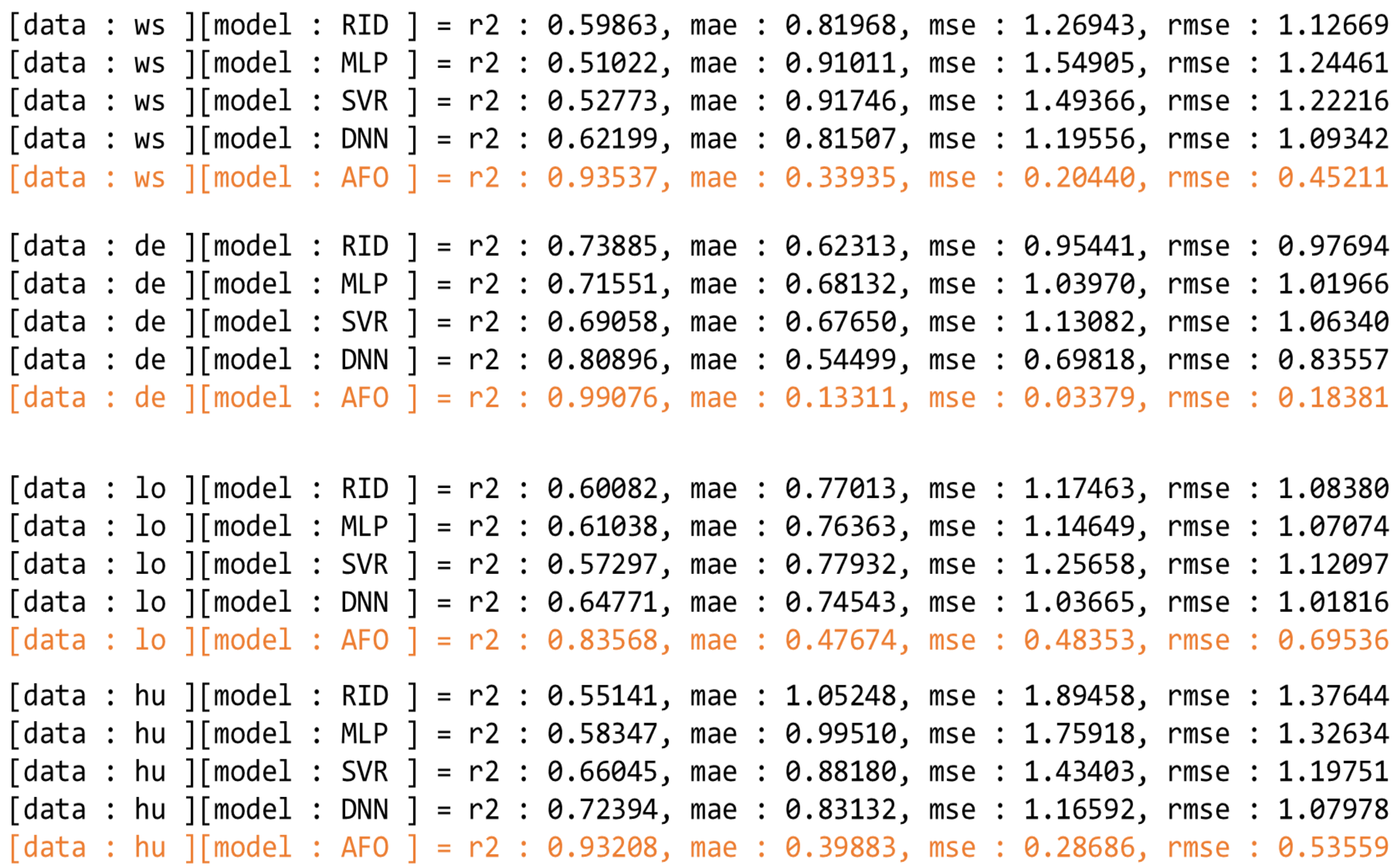
[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry



[Result and detail] Advanced model prediction by hyperparameter optimization in chemistry

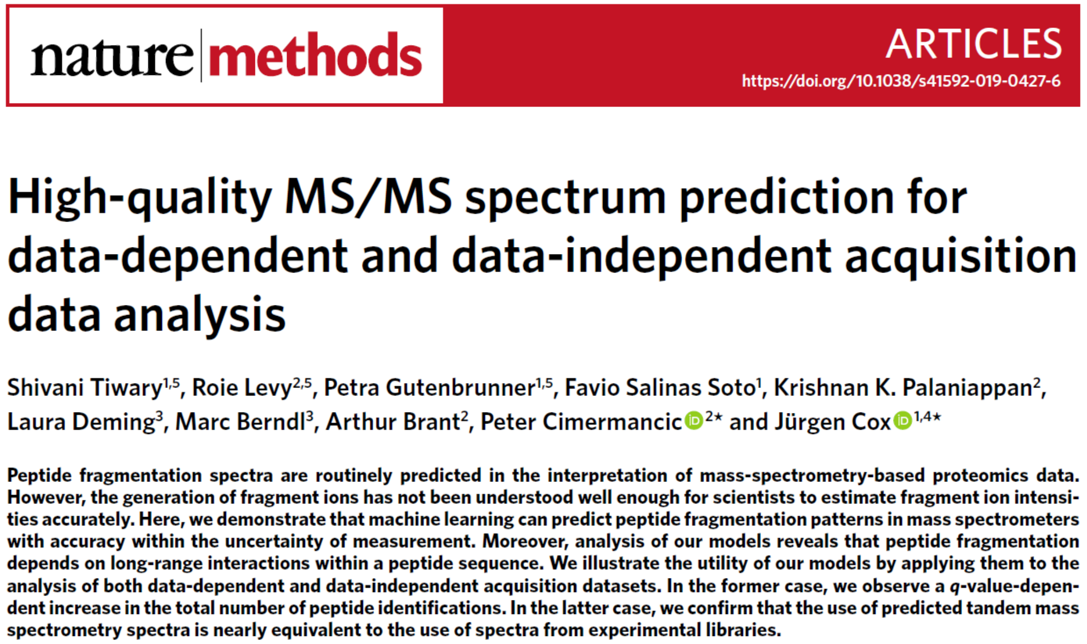
❖ Check the improvement by individual chemical descriptors.

- **Result 1:** the model prediction without no-feature engineering showed low performance.
- **Result 2:** Using the AFO method, it proved that the performance of the prediction improved the score than other prediction algorithm.
- **Result 3:** Following the result of the dataset prediction, it predicted high scores at stabilized dataset such as Delaney(de), ws496 (ws), Huuskonen(hu). However, the protein size molecule dataset such as lovric(lo) is necessary to customize the inputs and models.



```
[data : ws ][model : RID ] = r2 : 0.59863, mae : 0.81968, mse : 1.26943, rmse : 1.12669
[data : ws ][model : MLP ] = r2 : 0.51022, mae : 0.91011, mse : 1.54905, rmse : 1.24461
[data : ws ][model : SVR ] = r2 : 0.52773, mae : 0.91746, mse : 1.49366, rmse : 1.22216
[data : ws ][model : DNN ] = r2 : 0.62199, mae : 0.81507, mse : 1.19556, rmse : 1.09342
[data : ws ][model : AFO ] = r2 : 0.93537, mae : 0.33935, mse : 0.20440, rmse : 0.45211

[data : de ][model : RID ] = r2 : 0.73885, mae : 0.62313, mse : 0.95441, rmse : 0.97694
[data : de ][model : MLP ] = r2 : 0.71551, mae : 0.68132, mse : 1.03970, rmse : 1.01966
[data : de ][model : SVR ] = r2 : 0.69058, mae : 0.67650, mse : 1.13082, rmse : 1.06340
[data : de ][model : DNN ] = r2 : 0.80896, mae : 0.54499, mse : 0.69818, rmse : 0.83557
[data : de ][model : AFO ] = r2 : 0.99076, mae : 0.13311, mse : 0.03379, rmse : 0.18381


[data : lo ][model : RID ] = r2 : 0.60082, mae : 0.77013, mse : 1.17463, rmse : 1.08380
[data : lo ][model : MLP ] = r2 : 0.61038, mae : 0.76363, mse : 1.14649, rmse : 1.07074
[data : lo ][model : SVR ] = r2 : 0.57297, mae : 0.77932, mse : 1.25658, rmse : 1.12097
[data : lo ][model : DNN ] = r2 : 0.64771, mae : 0.74543, mse : 1.03665, rmse : 1.01816
[data : lo ][model : AFO ] = r2 : 0.83568, mae : 0.47674, mse : 0.48353, rmse : 0.69536

[data : hu ][model : RID ] = r2 : 0.55141, mae : 1.05248, mse : 1.89458, rmse : 1.37644
[data : hu ][model : MLP ] = r2 : 0.58347, mae : 0.99510, mse : 1.75918, rmse : 1.32634
[data : hu ][model : SVR ] = r2 : 0.66045, mae : 0.88180, mse : 1.43403, rmse : 1.19751
[data : hu ][model : DNN ] = r2 : 0.72394, mae : 0.83132, mse : 1.16592, rmse : 1.07978
[data : hu ][model : AFO ] = r2 : 0.93208, mae : 0.39883, mse : 0.28686, rmse : 0.53559
```

[Result and detail] Mass-Spectrum



nature methods

ARTICLES

https://doi.org/10.1038/s41592-019-0427-6

High-quality MS/MS spectrum prediction for data-dependent and data-independent acquisition data analysis

Shivani Tiwary[1,5], Roie Levy[2,5], Petra Gutenbrunner[1,5], Favio Salinas Soto[1], Krishnan K. Palaniappan[2], Laura Deming[3], Marc Berndl[3], Arthur Brant[2], Peter Cimermancic[2]* and Jürgen Cox[1,4]*

Peptide fragmentation spectra are routinely predicted in the interpretation of mass-spectrometry-based proteomics data. However, the generation of fragment ions has not been understood well enough for scientists to estimate fragment ion intensities accurately. Here, we demonstrate that machine learning can predict peptide fragmentation patterns in mass spectrometers with accuracy within the uncertainty of measurement. Moreover, analysis of our models reveals that peptide fragmentation depends on long-range interactions within a peptide sequence. We illustrate the utility of our models by applying them to the analysis of both data-dependent and data-independent acquisition datasets. In the former case, we observe a *q*-value-dependent increase in the total number of peptide identifications. In the latter case, we confirm that the use of predicted tandem mass spectrometry spectra is nearly equivalent to the use of spectra from experimental libraries.

❖ **Where are to use?**

- Amino acid sequence analysis of proteins and peptides
- Impurity evaluation during drug development
- Purity assessment of active pharmaceutical ingredients
- Regular analysis of illicit drugs in urine, blood and hair
- Genetic disease detection for amino acids, fatty acids, and organic biosynthesis

What is the Mass Spectrometry?

- An analytical technique that uses the mass-to-charge (m/z) ratio to identify chemicals in a sample
- For example, the sample from human blood, analyzing the mass-to-charge (m/z) to find out the potential appearance of cancers

- How?

the sample → gaseous ions → analyzing their mass-to-charge ration and **relative abundance (=intensity)**.

[Result and detail] Mass-Spectrum

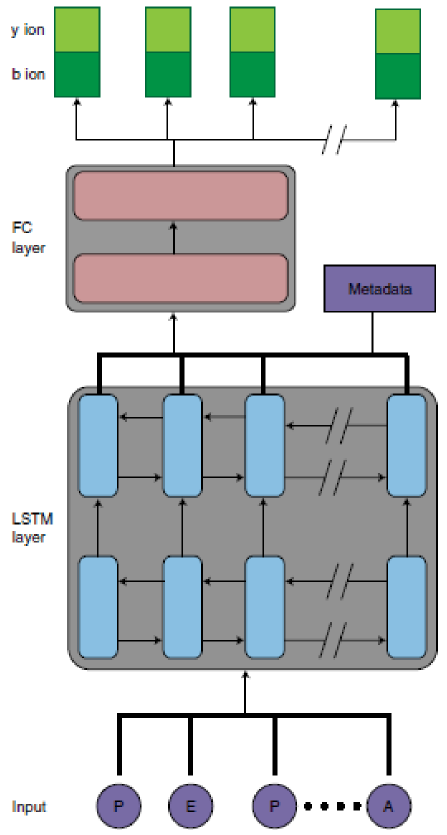
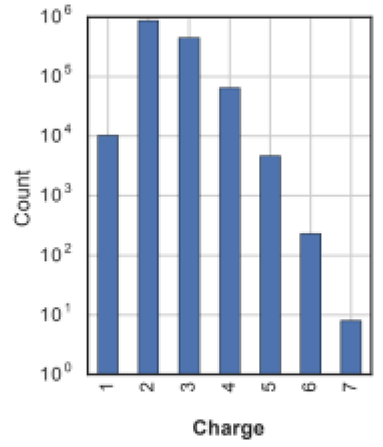


Fig. 1 | Bidirectional RNN architecture for the prediction of fragment intensities. The neural network contains two basic modules: an RNN encoder and a perceptron decoder. The encoder takes a one-hot-encoded peptide sequence as an input and outputs its fixed-length representation vector. The sequence representation vector is then combined with metadata features and input into the decoder. The decoder contains a set of fully connected layers that outputs intensities of different fragment ion types (for example, y and b ions) at each position in the input peptide sequence. FC, fully connected.

- **Predicting mass spectra limited**
  - reason 1. peptide sequences vary in length and incompatible with other algorithm because of their customized fixed-length input
  - reason 2. different fragmentation and acquisition method (=analyzing the gaseous ions) generates different results

- **What is the advantages by using the deep learning?**
  - ✓ RNNs are designed to work with sequential information and accepted with inputs with different levels (such as amino acid, peptide fragment and machine type) and types (amino acid identities or their physicochemical properties)
  - ✓ if the number of charges from ions higher (=complexity is higher), it will be difficult to analyze or predict the relative intensities (=y truth) through experimentation. However, using the deep learning can handle this problem with saving the cost and time.

[Result and detail] Mass-Spectrum

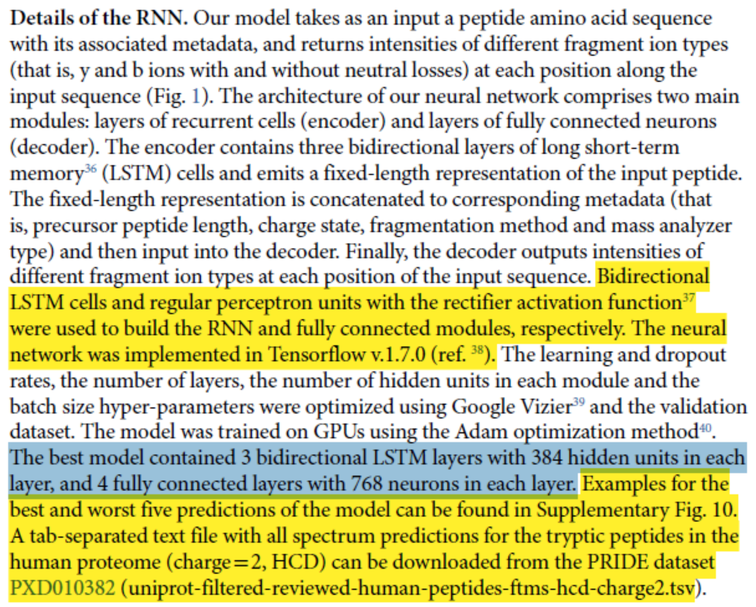


❖ Problem (Implementation)

- No code available, but only dataset
- Paper used the different metric, ‘PCC (Pearson Correlation coefficient)’
- Only 2 charge, but paper used 7 charge.
- The dataset count = 565460 (= too much time to training or testing)
  - ✓ each model training estimated time : 15 – 16hr (Nvidia GTX1060, 8GB)

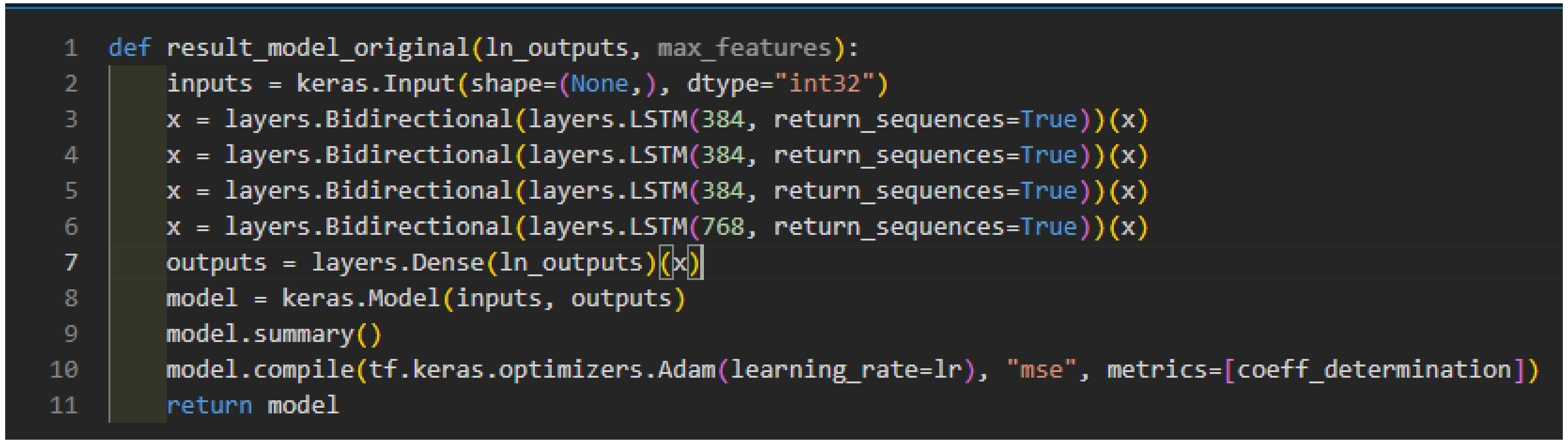
❖ Alternative way (Implementation)

- Code implementation
- Use the R2 score to nature architecture and try to improve the current version
- Using the AFO method



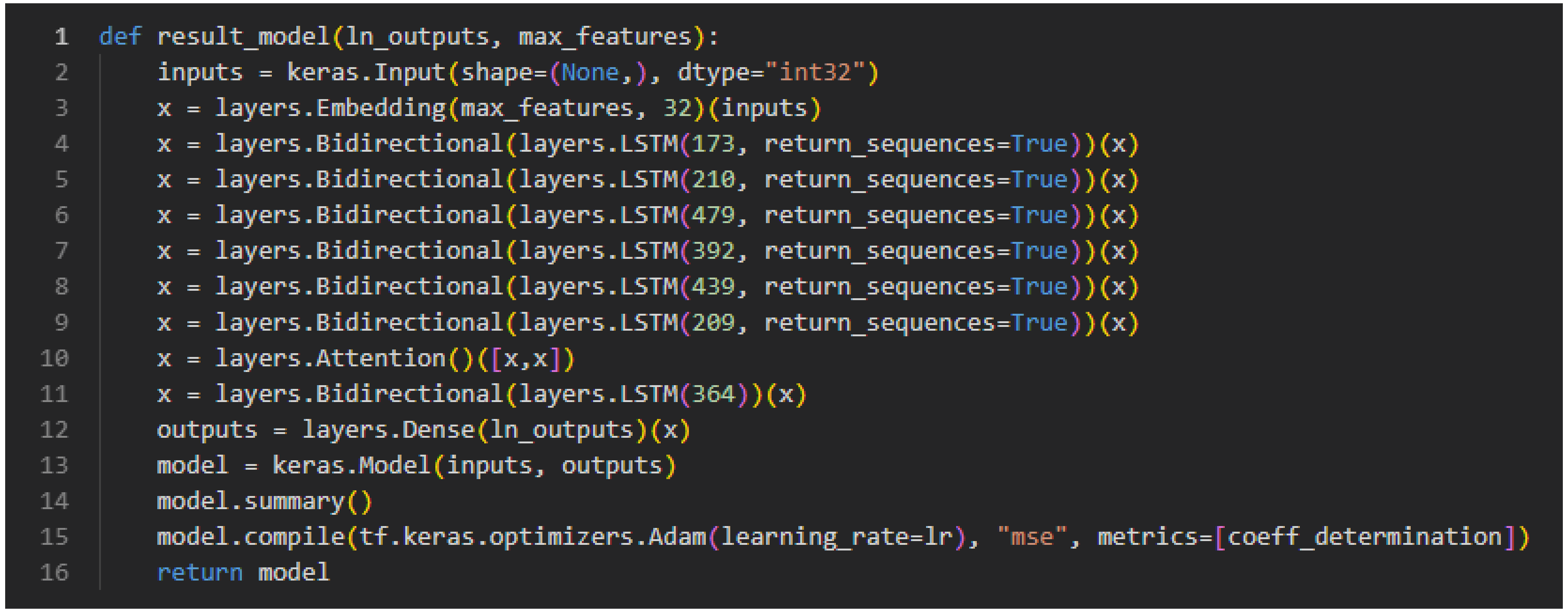
Details of the RNN. Our model takes as an input a peptide amino acid sequence with its associated metadata, and returns intensities of different fragment ion types (that is, y and b ions with and without neutral losses) at each position along the input sequence (Fig. 1). The architecture of our neural network comprises two main modules: layers of recurrent cells (encoder) and layers of fully connected neurons (decoder). The encoder contains three bidirectional layers of long short-term memory[36] (LSTM) cells and emits a fixed-length representation of the input peptide. The fixed-length representation is concatenated to corresponding metadata (that is, precursor peptide length, charge state, fragmentation method and mass analyzer type) and then input into the decoder. Finally, the decoder outputs intensities of different fragment ion types at each position of the input sequence. Bidirectional LSTM cells and regular perceptron units with the rectifier activation function[37] were used to build the RNN and fully connected modules, respectively. The neural network was implemented in Tensorflow v.1.7.0 (ref. [38]). The learning and dropout rates, the number of layers, the number of hidden units in each module and the batch size hyper-parameters were optimized using Google Vizier[39] and the validation dataset. The model was trained on GPUs using the Adam optimization method[40]. The best model contained 3 bidirectional LSTM layers with 384 hidden units in each layer, and 4 fully connected layers with 768 neurons in each layer. Examples for the best and worst five predictions of the model can be found in Supplementary Fig. 10. A tab-separated text file with all spectrum predictions for the tryptic peptides in the human proteome (charge = 2, HCD) can be downloaded from the PRIDE dataset PXD010382 (uniprot-filtered-reviewed-human-peptides-ftms-hcd-charge2.tsv).

Original



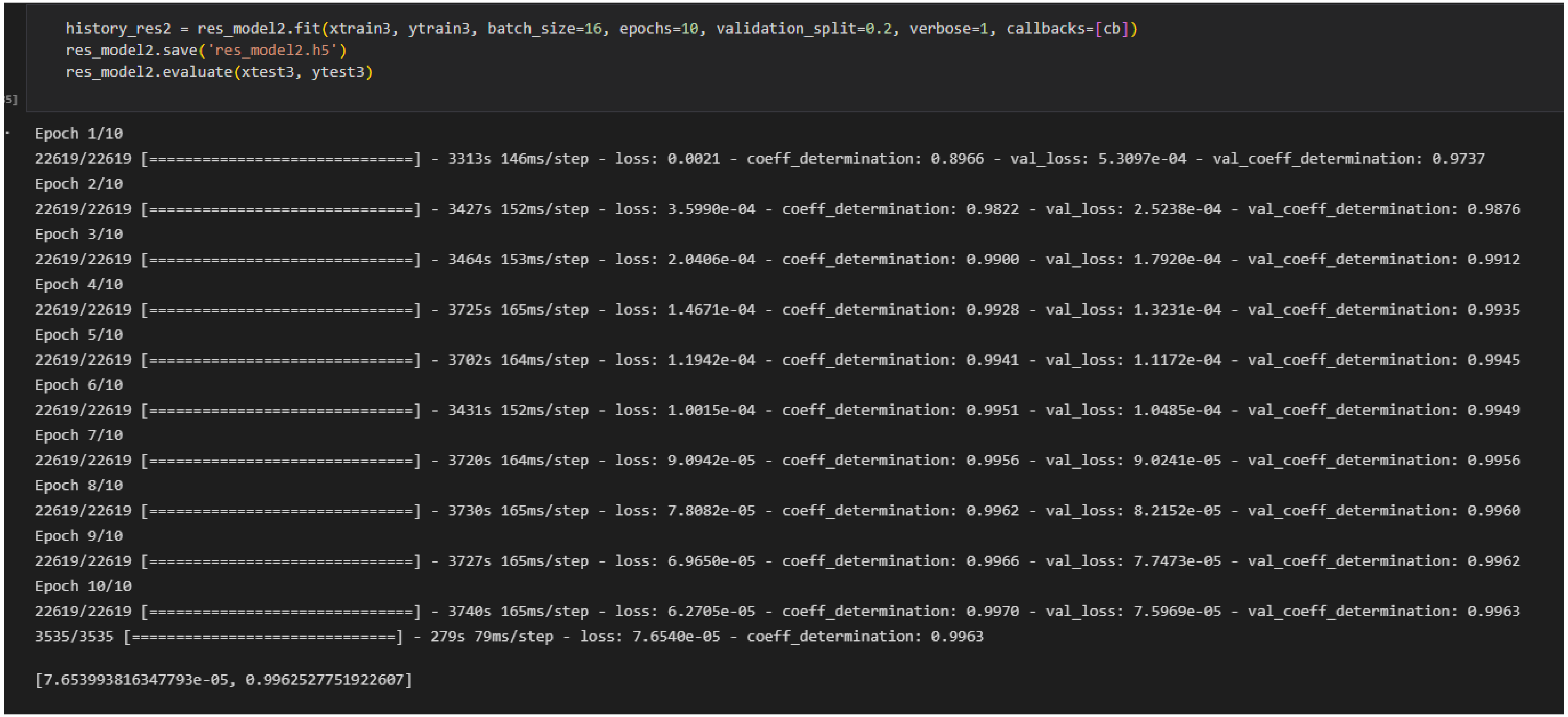
```python
def result_model_original(ln_outputs, max_features):
    inputs = keras.Input(shape=(None,), dtype="int32")
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(384, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(768, return_sequences=True))(x)
    outputs = layers.Dense(ln_outputs)(x)
    model = keras.Model(inputs, outputs)
    model.summary()
    model.compile(tf.keras.optimizers.Adam(learning_rate=lr), "mse", metrics=[coeff_determination])
    return model
```

Improvement (Hyperparameter optimization + Attention)



```python
def result_model(ln_outputs, max_features):
    inputs = keras.Input(shape=(None,), dtype="int32")
    x = layers.Embedding(max_features, 32)(inputs)
    x = layers.Bidirectional(layers.LSTM(173, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(210, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(479, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(392, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(439, return_sequences=True))(x)
    x = layers.Bidirectional(layers.LSTM(209, return_sequences=True))(x)
    x = layers.Attention()([x,x])
    x = layers.Bidirectional(layers.LSTM(364))(x)
    outputs = layers.Dense(ln_outputs)(x)
    model = keras.Model(inputs, outputs)
    model.summary()
    model.compile(tf.keras.optimizers.Adam(learning_rate=lr), "mse", metrics=[coeff_determination])
    return model
```

❖ Mass-Spectrum Prediction



```
history_res2 = res_model2.fit(xtrain3, ytrain3, batch_size=16, epochs=10, validation_split=0.2, verbose=1, callbacks=[cb])
res_model2.save('res_model2.h5')
res_model2.evaluate(xtest3, ytest3)
```

```
Epoch 1/10
22619/22619 [==============================] - 3313s 146ms/step - loss: 0.0021 - coeff_determination: 0.8966 - val_loss: 5.3097e-04 - val_coeff_determination: 0.9737
Epoch 2/10
22619/22619 [==============================] - 3427s 152ms/step - loss: 3.5990e-04 - coeff_determination: 0.9822 - val_loss: 2.5238e-04 - val_coeff_determination: 0.9876
Epoch 3/10
22619/22619 [==============================] - 3464s 153ms/step - loss: 2.0406e-04 - coeff_determination: 0.9900 - val_loss: 1.7920e-04 - val_coeff_determination: 0.9912
Epoch 4/10
22619/22619 [==============================] - 3725s 165ms/step - loss: 1.4671e-04 - coeff_determination: 0.9928 - val_loss: 1.3231e-04 - val_coeff_determination: 0.9935
Epoch 5/10
22619/22619 [==============================] - 3702s 164ms/step - loss: 1.1942e-04 - coeff_determination: 0.9941 - val_loss: 1.1172e-04 - val_coeff_determination: 0.9945
Epoch 6/10
22619/22619 [==============================] - 3431s 152ms/step - loss: 1.0015e-04 - coeff_determination: 0.9951 - val_loss: 1.0485e-04 - val_coeff_determination: 0.9949
Epoch 7/10
22619/22619 [==============================] - 3720s 164ms/step - loss: 9.0942e-05 - coeff_determination: 0.9956 - val_loss: 9.0241e-05 - val_coeff_determination: 0.9956
Epoch 8/10
22619/22619 [==============================] - 3730s 165ms/step - loss: 7.8082e-05 - coeff_determination: 0.9962 - val_loss: 8.2152e-05 - val_coeff_determination: 0.9960
Epoch 9/10
22619/22619 [==============================] - 3727s 165ms/step - loss: 6.9650e-05 - coeff_determination: 0.9966 - val_loss: 7.7473e-05 - val_coeff_determination: 0.9962
Epoch 10/10
22619/22619 [==============================] - 3740s 165ms/step - loss: 6.2705e-05 - coeff_determination: 0.9970 - val_loss: 7.5969e-05 - val_coeff_determination: 0.9963
3535/3535 [==============================] - 279s 79ms/step - loss: 7.6540e-05 - coeff_determination: 0.9963

[7.653993816347793e-05, 0.9962527751922607]
```

R2 score evaluation (with test data) : **0.99625**