Electronic Supplementary Material (ESI) for Journal of Materials Chemistry A.

Supplementary Information for

Machine Learning Enabled Exploration of Multicomponent Metal Oxides for Catalyzing Oxygen Reduction in Alkaline Media

Xue Jia* and Hao Li*

Advanced Institute for Materials Research (WPI-AIMR), Tohoku University, Sendai 980-8577, Japan

*** Corresponding Author:**

jia.xue.d8@tohoku.ac.jp (X. J.)

li.hao.b8@tohoku.ac.jp (H. L.)

ML algorithms

The coefficient of determination (R^2) and root mean squared error (RMSE) were used to evaluate the performance of the ML model. A higher R^2 value (*i.e.*, the value closer to 1.0), or a lower RMSE (*i.e.*, the value closer to 0 $\mu A \cdot cm^{-2}$), signifies a better model performance. It is important to note that when using ANN, the features should be scaled to have a mean of zero and a standard deviation of one [1]. Conversely, XGBoost and LightGBM, as tree-based ensemble models, do not require such scaling, as they primarily focus on the distribution between variables rather than their absolute scale [2].

We first used ANN to build a model based on the initially whole dataset to evaluate the data quality. For the final dataset after cleaning, we employed ANN, XGBoost, and LightGBM to build the ML models on the training dataset and compare their performance. The hyperparameters for the models built based on the dataset in 0.8 and 0.63 V_{RHE} are respectively shown in **Tables S1** and S**2**.

Table S1 The hyperparameters for different algorithms based on the dataset in 0.8 V_{RHE}.

Initial dataset	**Final dataset after cleaning**		
ANN	**ANN**	**XGBoost**	**LightGBM**
hidden_layer_sizes = (150,100,40) activation = 'relu' solver = 'adam' alpha = 0.01 learning_rate_init = 0.01 max_iter = 100 batch_size = 64 random_state = 42 verbose = 1	hidden_layer_sizes = (150,100,60) activation = 'relu' solver = 'adam' alpha = 0.01 learning_rate_init = 0.01 max_iter = 100 batch_size = 64 random_state = 42 verbose = 1	n_estimators = 500 colsample_bytree = 0.8 gamma = 0.0 max_depth = 6 min_child_weight = 2 reg_alpha = 1 reg_lambda = 1 subsample = 1 learning_rate=0.01 random_state = 0 n_jobs = 2	n_estimators = 500 boosting_type = 'gbdt' objective = 'regression' colsample_bytree = 0.6 max_depth = 9 min_child_samples = 3 num_leaves = 13 subsample = 0.2 learning_rate = 0.01 reg_alpha = 0.1 reg_lambda = 0.1 random_state = 42 force_col_wise=True is_unbalance = True verbose = -1

Table S2 The hyperparameters for different algorithms based on the dataset in 0.63 V_{RHE}.

Initial dataset	Final dataset after cleaning		
ANN	**ANN**	**XGBoost**	**LightGBM**
hidden_layer_sizes = (200,150,70) activation = 'relu' solver = 'adam' alpha = 0.01 learning_rate_init = 0.01 max_iter = 100 batch_size = 64 random_state = 42 verbose = 1	hidden_layer_sizes = (200,100,50) activation = 'relu' solver = 'adam' alpha = 0.01 learning_rate_init = 0.01 max_iter = 200 batch_size = 64 random_state = 42 verbose = 1	n_estimators = 500 colsample_bytree = 0.8 gamma = 0.0 max_depth =7 min_child_weight = 3 reg_alpha = 1 reg_lambda = 1 subsample = 1 learning_rate = 0.01 random_state = 0 n_jobs = 2	n_estimators = 500 boosting_type = 'gbdt' objective = 'regression' colsample_bytree = 0.6 max_depth = 8 min_child_samples = 2 num_leaves = 13 subsample = 0.2 learning_rate = 0.01 reg_alpha = 0.1 reg_lambda = 0.1 random_state = 42 force_col_wise = True is_unbalance = True verbose = -1

Furthermore, we performed the symbolic regression model using the elemental property features in **Figure 2c-d**. The mode hypterparameters are: populations = 30, model_selection = 'best', niterations = 50, binary_operators = ['+', '-', '*', '/'], unary_operators=['cos', 'exp', 'sin', 'neg', 'square', 'log10', 'tan']. The hyperparameters for ROOST and CrabNet are default parameters.

Table S3 The equations generated by symbolic regression based on the dataset in 0.80 V_{RHE}. The features x_0, x_1, ..., x_{15} correspond to the features shown in **Figure 2c-d**.

	Score	Equation
0	0	-0.86954945
1	0.08	-24.075083 / x_{12}
2	0.01	-0.6569974 / cos(x_{11})
3	0.13	(x_{12} / x_5) + -1.518347
4	0.02	tan(-1.2127134 + sin(x_{12} / x_5))
5	0.02	tan(sin(sin(x_{12} / x_5)) + -1.1909018)
6	0.12	-1.4982516 + sin(0.111019574 * (x_{10} + tan(x_{10})))
7	0.12	tan(sin(0.10763502 * (tan(x_{10}) + x_{10})) - x_3)
8	0.10	tan(sin(sin(0.10763502 * (tan(x_{10}) + x_{10}))) - x_3)
9	0.03	tan(sin(sin(sin(0.10763502 * (x_{10} + tan(x_{10}))))) + -1.1909018)
10	0.02	(x_{11} * -0.4162485) + tan(sin((tan(x_{10}) + x_{10}) * 0.10763502) - x_3)
11	0.04	tan(sin((x_{10} + tan(x_{10})) * square(-0.28647247)) - 0.8872833) - cos(cos(x_{11}))
12	0.05	(tan(sin((tan(x_{10}) + x_{10}) * 0.10763502) - x_3) / x_3) + (-0.44395217 * x_{11})
13	0.07	(tan(sin(sin((tan(x_{10}) + x_{10}) * 0.10763502)) - x_3) / x_3) + (x_{11} * -0.44395217)
14	0.00	(tan(sin(sin((tan(x_{10}) + x_{10}) * sin(0.10763502))) - x_3) / x_3) + (x_{11} * -0.44395217)
15	0.01	(x_{11} * -0.44395217) + (tan(sin(sin(((tan(x_{10}) + x_{10}) * 0.10763502) + x_4)) - x_3) / x_3)
16	0.00	(x_{11} * sin(-0.44395217)) + (tan(sin(sin(((tan(x_{10}) + x_{10}) * 0.10763502) + x_4)) - x_3) / x_3)

Table S4 The equations generated by symbolic regression based on the dataset in 0.63 V_{RHE}. The features x_0, x_1, ..., x_{15} correspond to the features shown in **Figure 2c-d**.

	Score	Equation
0	0	-0.31296986
1	0.03	x_{11} * -0.50815225
2	0.16	sin(-11.425274 / x_{12})
3	0.13	sin(square(tan(x_{10})) * -0.30022746)
4	0.20	sin(tan(square(tan(x_{10})) * -0.30305976))
5	0.07	tan(sin(tan(square(tan(x_{10})) * -0.30305976)))
6	0.01	x_3 * sin(tan(-0.30305976 * square(tan(x_{10}))))
7	0.04	tan(sin(tan(-0.30305976 * square(tan(x_{10}))))) / 1.1619223
8	0.05	tan(neg(square(cos(square(square(square(tan(x_{10})))) - x_3))))
9	0.01	tan(neg(square(square(cos(square(square(square(tan(x_{10})))) - x_3)))))
10	0.04	tan(neg(square(square(cos(sin(square(square(square(tan(x_{10})))) - x_3))))))
11	0.02	tan(neg(square(square(cos((x_3 - square(square(square(tan(x_{10}))))) + -0.13868064)))))
12	0.00	tan(neg(square(square(cos((x_3 - square(square(square(tan(x_{10}))))) + tan(-0.13868064))))))
13	0.02	tan(neg(square(square(cos(x_3 - (square(square(square(tan(x_{10})))) - -0.121624485)))))) - 0.048299428
14	0.00	tan(neg(square(square(cos((square(square(square(tan(x_{10})))) - -0.121624485) - x_3))))) + neg(tan(0.048299428))
15	0.00	tan(neg(square(sin(square(sin(square(square(tan(square(tan(x_{10})))) - -1.7352135))) * x_3)) * x_3)) + -0.04448677

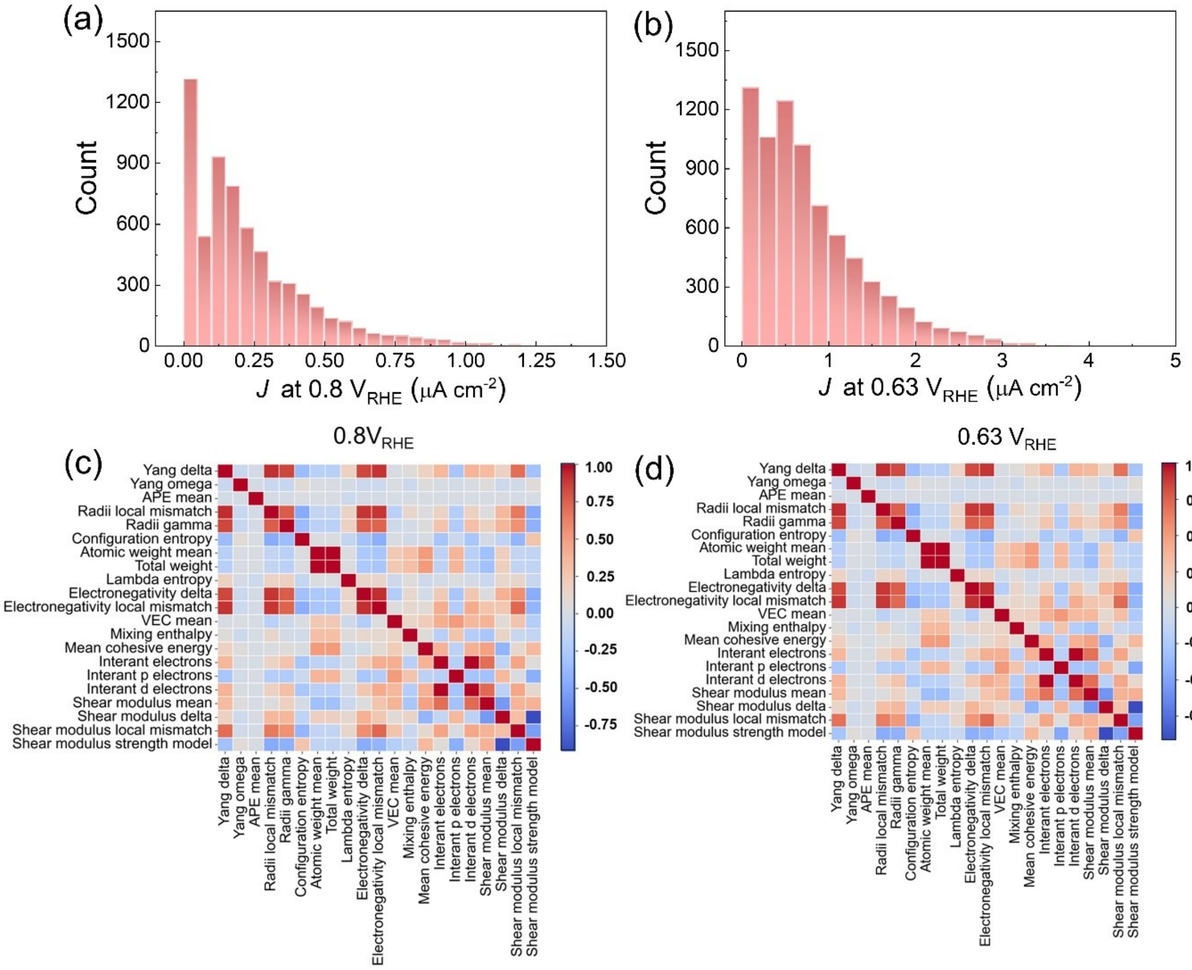

Figure S1 (a-b) Distribution of current densities under (a) 0.8 V_{RHE} and (b) 0.63 V_{RHE} with the original values. (c-d) Pearson correlations among 21 features in the (c) 0.8 V_{RHE} dataset and (d) 0.63 V_{RHE} dataset.

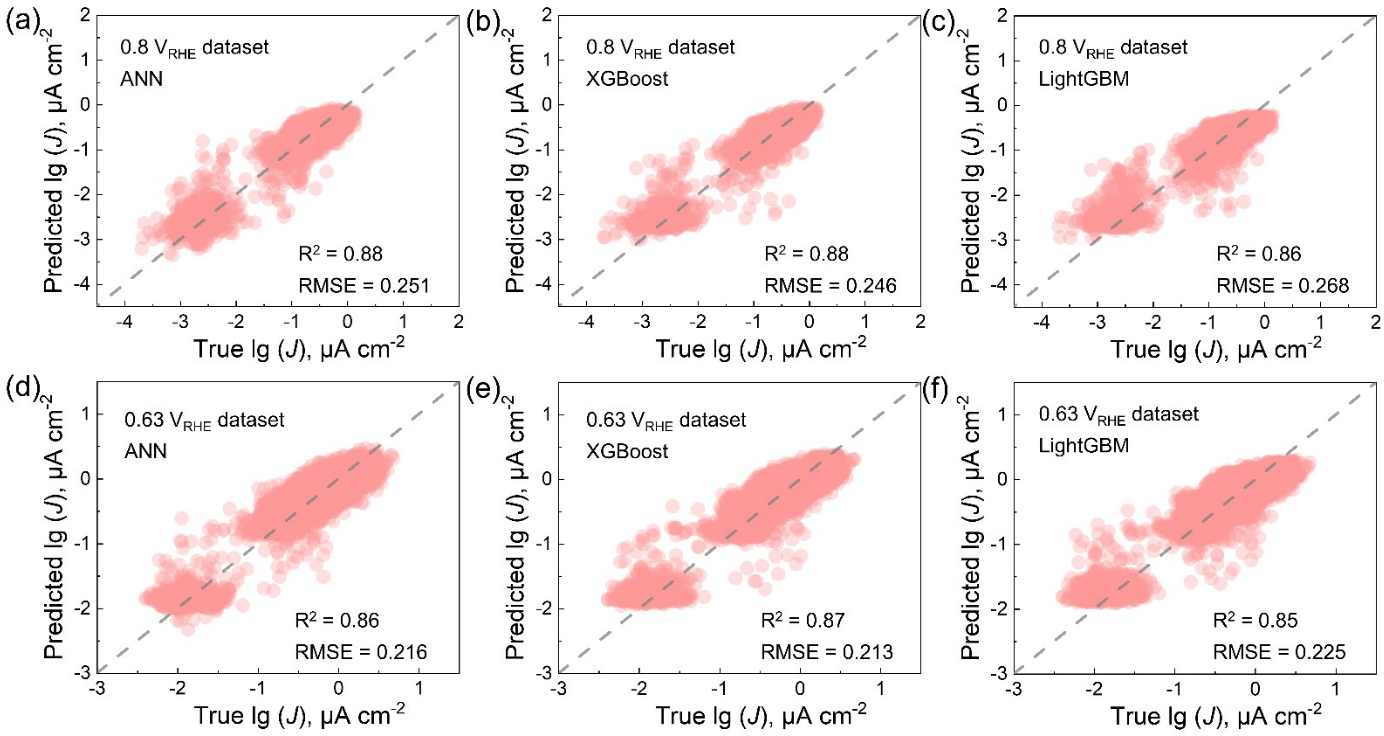

Figure S2 Performance of the models built by (a, d) ANN, (b, e) XGBoost, and (c, f) LightGBM on the 0.8 V_{RHE} and 0.63 V_{RHE} training dataset using 10-fold cross validation.

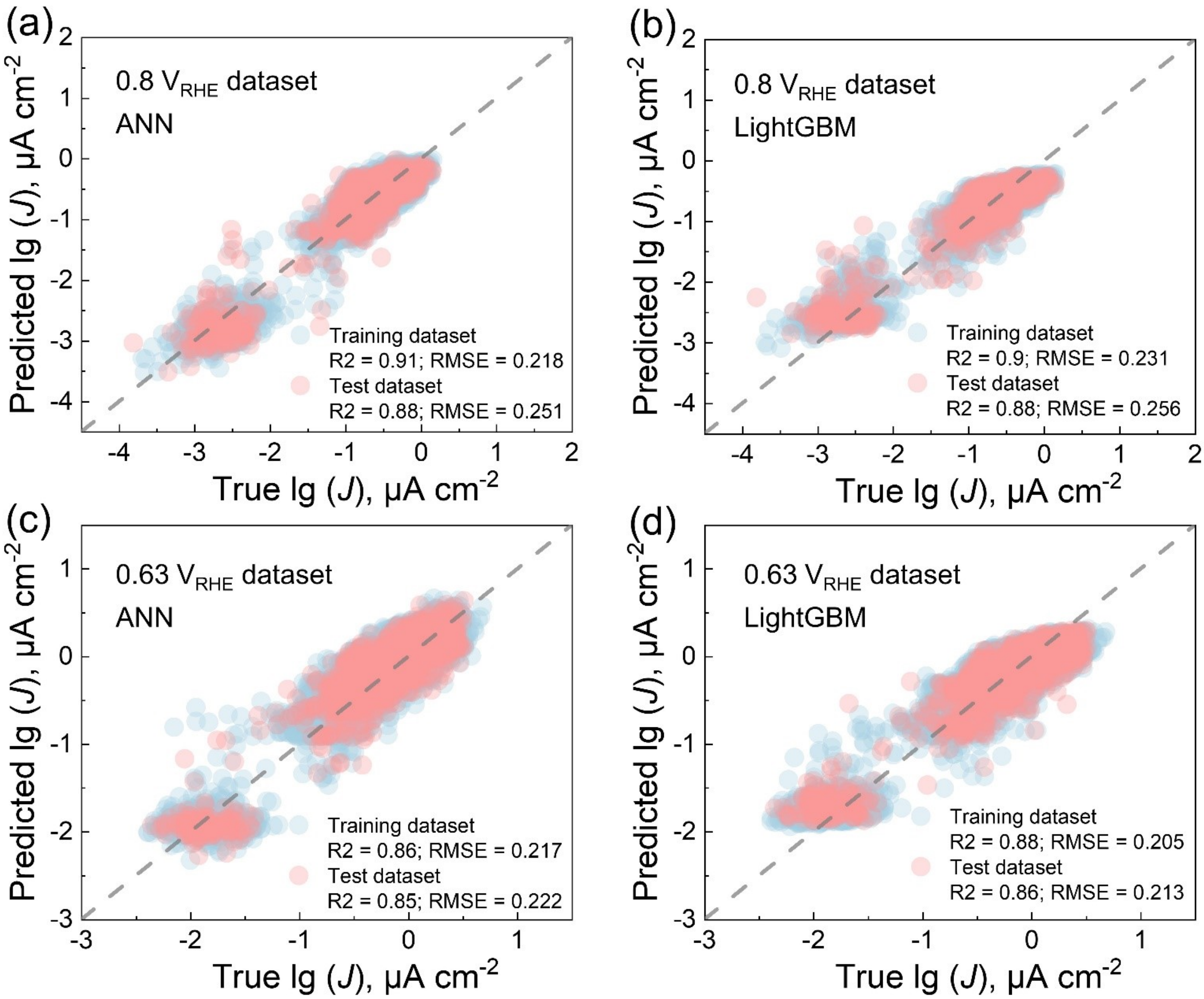

Figure S3 Performance of the models built by (a, c) ANN, and (b, d) LightGBM on the 0.8 V_{RHE} and 0.63 V_{RHE} training dataset and test dataset.

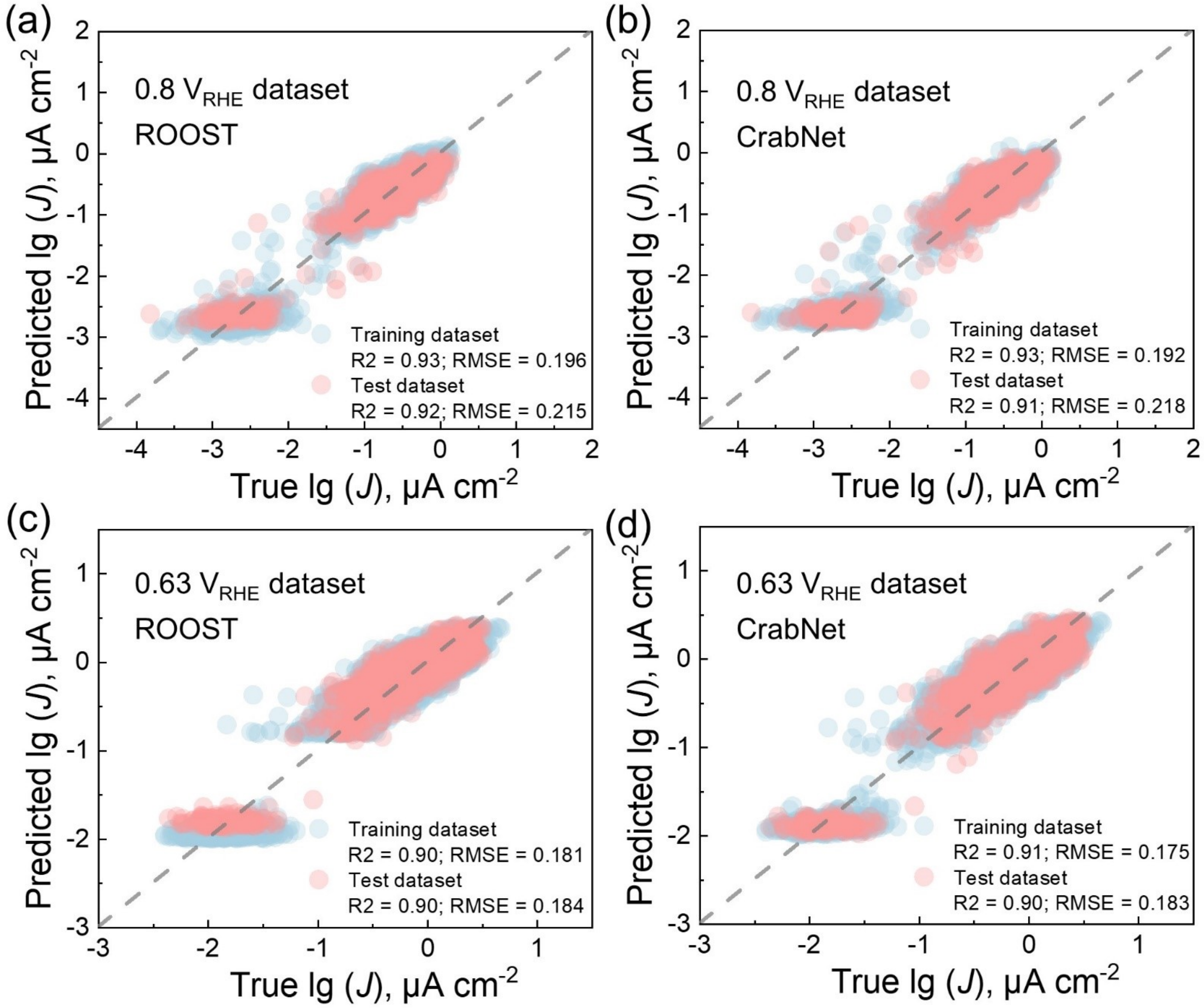

Figure S4 Comparison between experimental and predicted values by ROOST on the (a) training and (b) test sets, and by CrabNet on the (c) training and (d) test sets. The unit of RMSE is lg(μA·cm^{-2}).

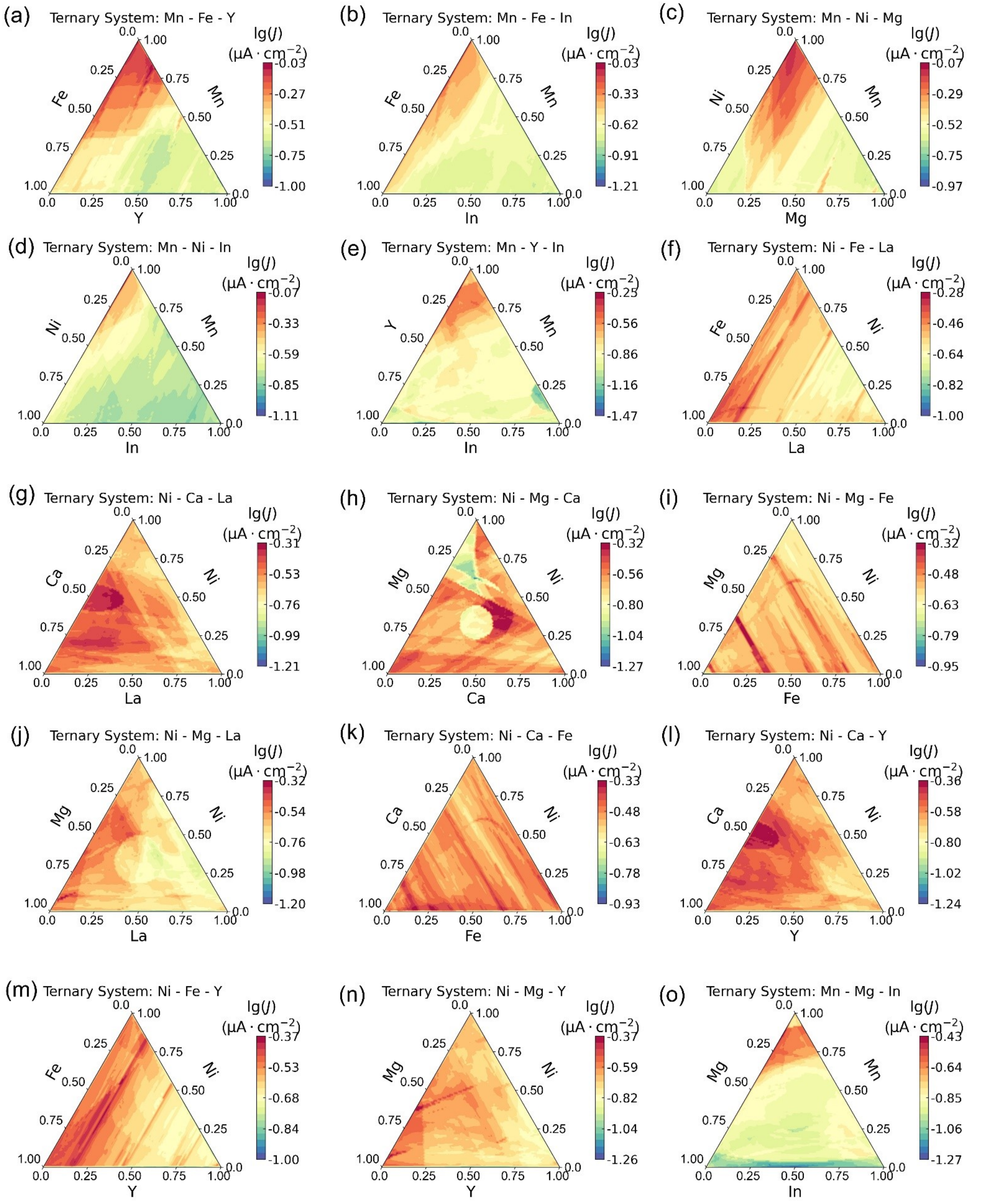

Figure S5 Predictive current density values at 0.8 V_{RHE} for the ternary systems based on the trained XGBoost model.

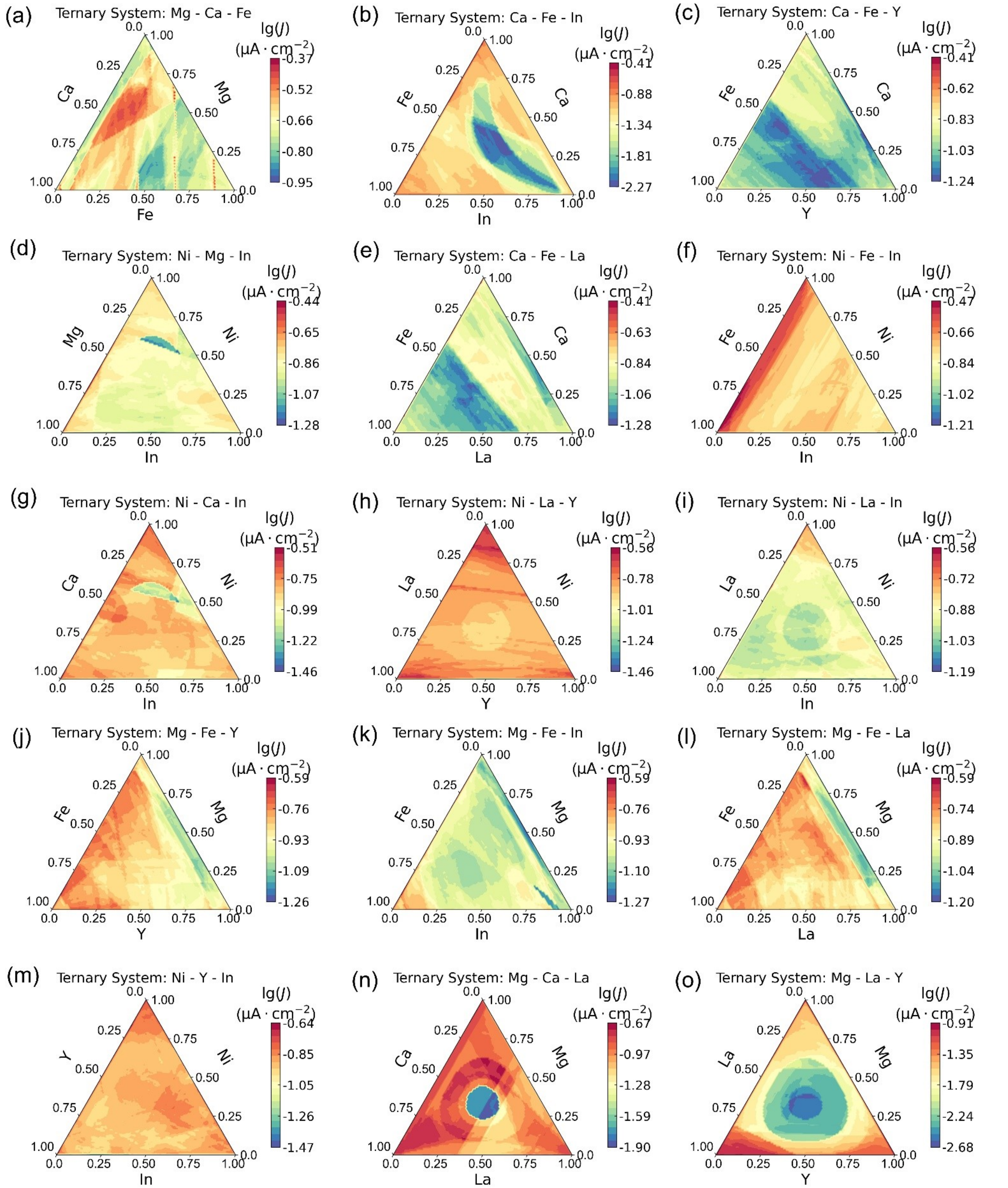

Figure S6 Predictive current density values at 0.8 V_{RHE} for the ternary systems based on the trained XGBoost model.

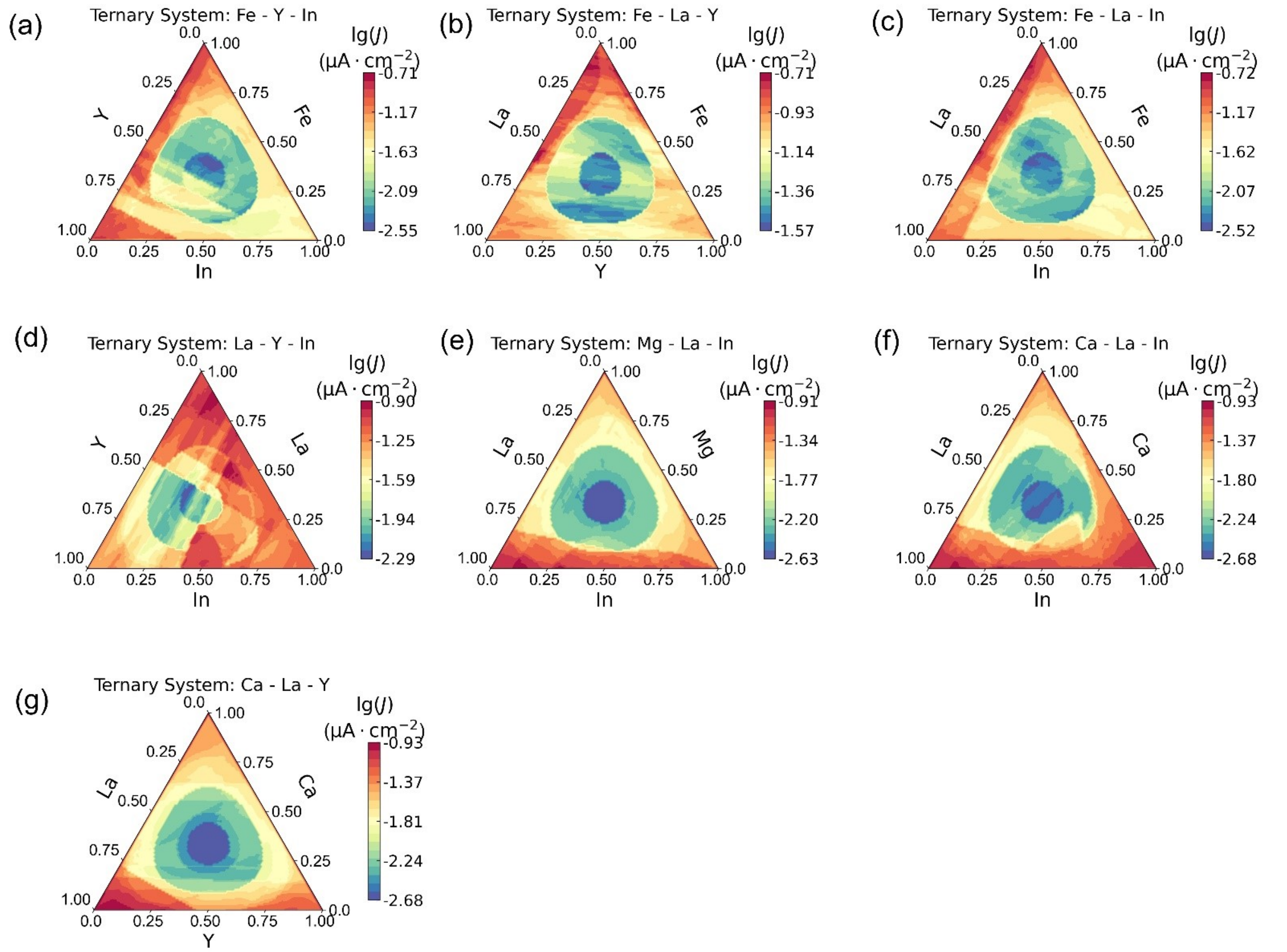

Figure S7 Predictive current density values at 0.8 V_{RHE} for the ternary systems based on the trained XGBoost model.

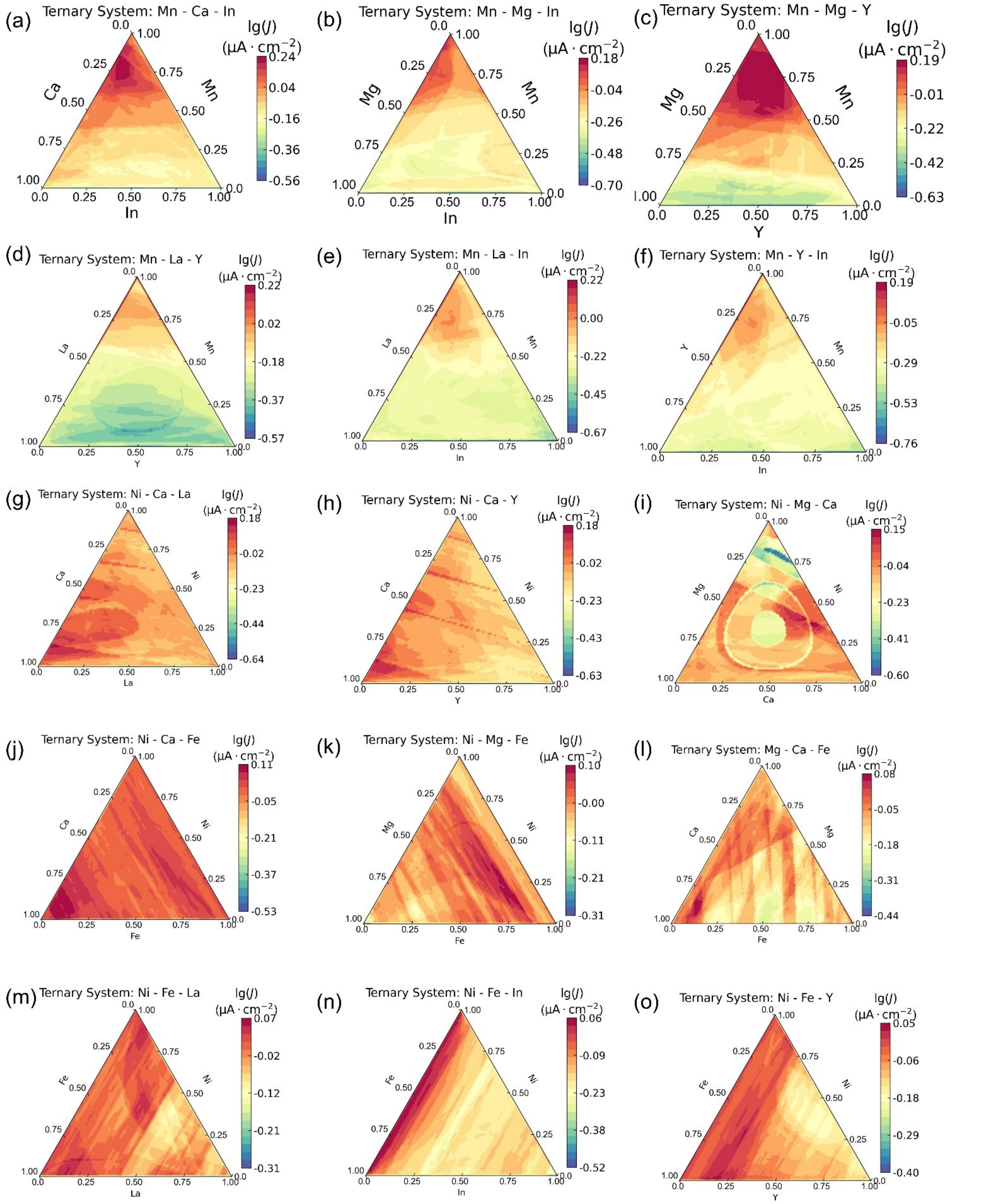

Figure S8 Predictive current density values at 0.63 V_{RHE} for the ternary systems based on the trained XGBoost model.

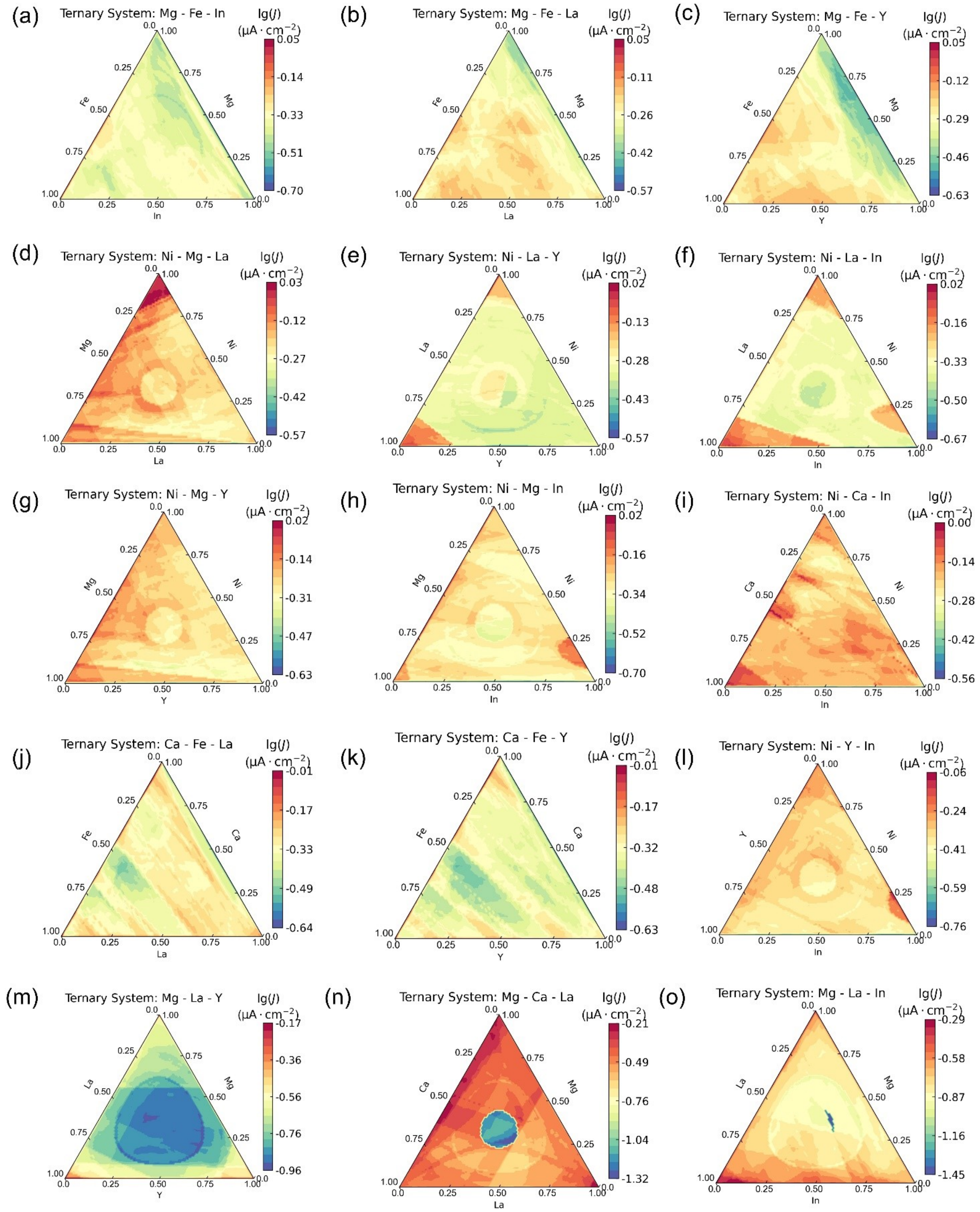

Figure S9 Predictive current density values at 0.63 V_{RHE} for the ternary systems based on the trained XGBoost model.

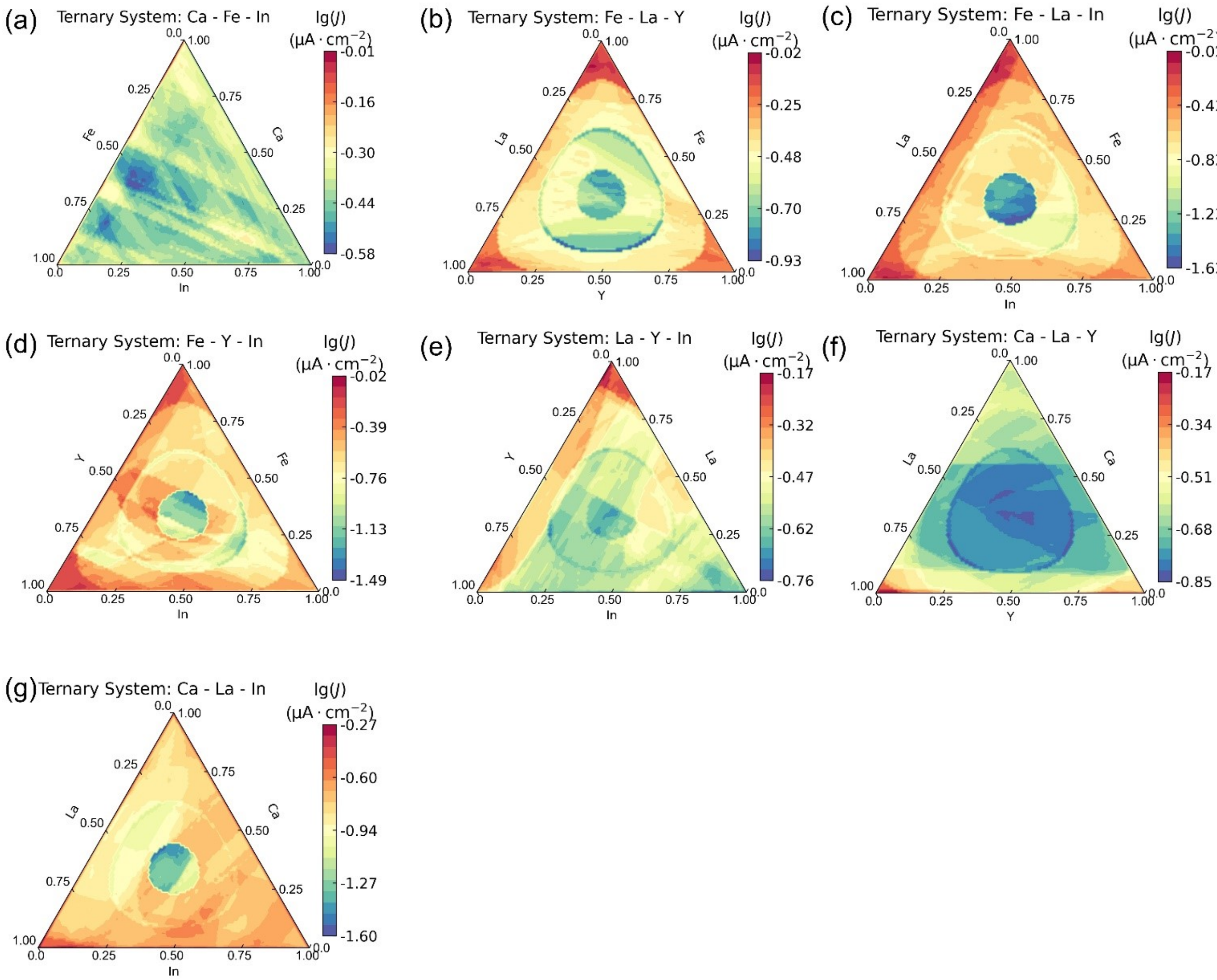

Figure S10 Predictive current density values at 0.63 V_{RHE} for the ternary systems based on the trained XGBoost model.

Reference

[1] L. Wang, K. Fu, Artificial Neural Networks, in: Wiley Encyclopedia of Computer Science and Engineering, 2009: pp. 181–188. https://doi.org/10.1002/9780470050118.ecse021.

[2] D. Sepiolo, A. Ligęza, Towards Explainability of Tree-Based Ensemble Models. A Critical Overview, in: W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak, J. Kacprzyk (Eds.), New Advances in Dependability of Networks and Systems, Springer International Publishing, Cham, 2022: pp. 287–296.