

Prediction of Thermodynamic Stability of Inorganic Crystals Using Machine Learning

Sara Nabili

August 8, 2025

Introduction

Purpose: This project uses a *supervised* Deep Neural Network (DNN) *binary classification* model to predict the thermodynamic stability of inorganic crystals based on various features.

Motivation: Predicting compound stability is costly and slow. This project aims to accelerate discovery using data-driven models.

Tools/Frameworks: *Python, scikit-learn, TensorFlow, XGBoost, pymatgen*, data-analysis toolkits, visualization packages, parallel and multiprocessing computing frameworks

Implementation: Feature extracted using pymatgen, engineered descriptors like bond statistics and atomic fractions, and trained ML algorithms

Final Result: DNN model achieved 87% validation accuracy and 94% discrimination power, showing promising results for early-stage stability screening.

Application: This tool can be used for pharmaceutical, sustainable energy technologies, and material engineering.

General Information

Problem Context: Thermodynamic stability (under a set of conditions) is achieved if the formation energy of the compound can not be lowered by rearranging its atoms [2].

Energy lowering mechanism:

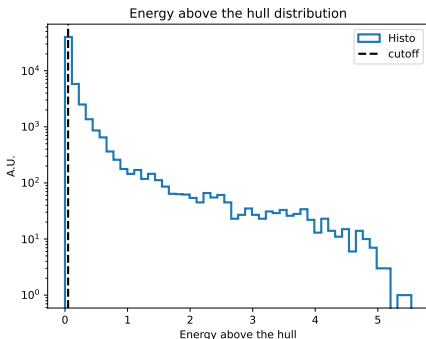
- ▶ *Decomposition:* Phase separation into competing materials that share an identical overall composition
- ▶ *Polymorphism:* Phase transition into an alternative crystal structure (polymorph) at fixed composition.

Thermodynamic Stability:

- ▶ A convex hull represents the lowest formation energy envelope derived from inorganic crystalline materials plotted against their composition.
- ▶ *The energy above the hull (EBH):* The energy difference between a compound and the convex hull with the same composition.

Energy above the hull

- ▶ The ML algorithms are trained to predict the thermodynamic stability of chemical compounds by the binary classification of the energy above the hull feature (EBH) [2]. The EBH distribution is shown below.
- ▶ In this study, thermodynamically stable to slightly metastable materials are classified by applying a threshold of $\text{EBH} \leq 0.05$ as suggested by [4].
- ▶ Out of nearly 50,000 materials in the dataset, roughly half meet the criteria for thermodynamic stability, while the other half fall above the defined threshold and are considered unstable.



Thermodynamic Stability: DFT & the Rationale for ML

Density Functional Theory [2] (DFT) is a powerful and widely used computational method for predicting the thermodynamic stability of materials and molecules. Its power lies in its excellent balance between accuracy and computational cost.

DFT's Limitations

- ▶ Highly accurate, but computationally intensive, especially for large or low symmetry structures.
- ▶ High-throughput screening of new compounds is limited by cost and runtime.
- ▶ Not suited for rapid prototyping or exploration of vast chemical spaces.

How Machine Learning Addresses These Bottlenecks:

- ▶ Learns structure property relationships from curated datasets.
- ▶ Predicts material stability orders of magnitude faster than DFT
- ▶ Enables scalable screening of millions of candidate materials

Study Limitations and Future Directions

Limitations:

- ▶ **Dataset Scale:** Access to the Materials Project database was limited to 50,000 compounds, constraining the deep neural network's generalization capacity.
- ▶ **Computational Resources:** Experiments were conducted on a single machine with 8 CPU cores, limiting parallelization and throughput.

Future Directions:

- ▶ Expanding access to larger datasets can improve model performance and diversity.
- ▶ Deploying high-performance clusters and distributed computing frameworks (e.g., HTCondor) will support advanced feature engineering and more complex ML architectures.

Data Collection

Data source: Data extracted from the Materials Project [6], an open database of DFT-computed material properties.

Feature extraction: The features are classified into four classes:

- ▶ **Energy & electronic features:** Helps capture energetic contribution
- ▶ **Structural and composition:** Gives access to size, packaging, and complexity
- ▶ **Bond features:** Facilitates the detection and quantification of inter-element bonding, while representing the surrounding chemical context
- ▶ **Atomic fraction:** Reflects elemental influence on stability

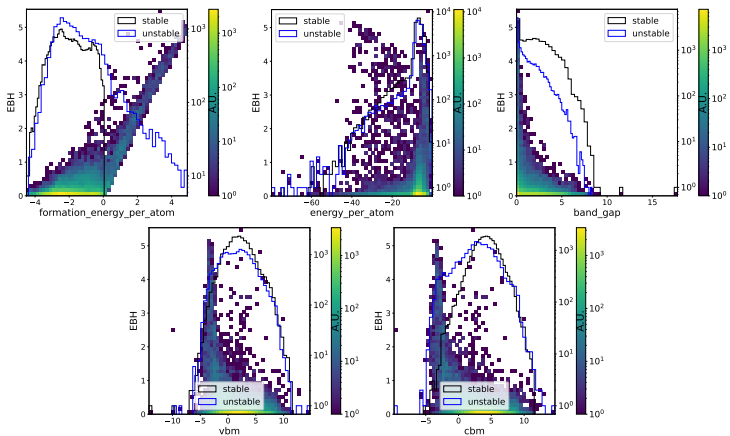
The first two classes are database queries; the rest need feature engineering. In total, 97 features were used as input to ML methods.

Feature Engineering To build:

- ▶ Atomic fractions: Computed element-wise from compositions.
- ▶ Bond structure statistics: Using neighborhood-finding algorithms, MinimumDistanceNN [8]:
 - ▶ Number of bonds
 - ▶ Mean bond length
 - ▶ Bond length standard deviation

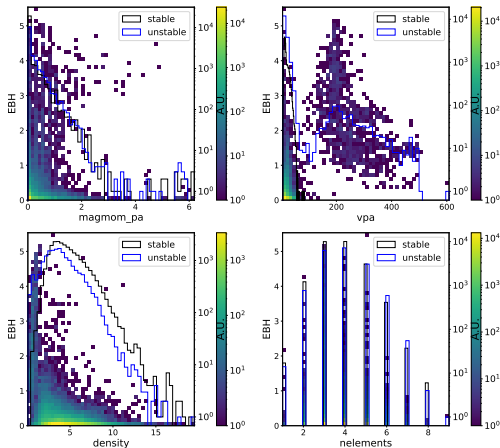
Feature: Energy & Electronic

- ▶ **Formation energy per atom:** Energy change as a compound is formed from its constituent elements in their standard states
- ▶ **Energy per atom:** include element reference in formation energy per atom
- ▶ **Band gap:** related to chemical bonding and electronic stability
- ▶ **Valence/conduction band edges:** explores patterns in electronic structure



Feature: Structure and Composition

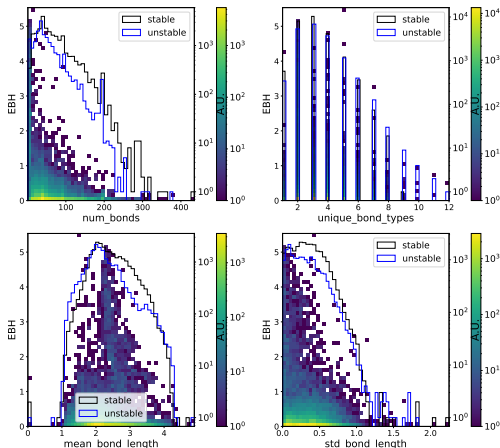
- ▶ **Magnetic moment per atom:** Depending on the material, some stable phases are magnetic
- ▶ **Volume per atom:** global structure feature
- ▶ **Density:** affected by atomic mass and volume
- ▶ **Number of elements and sites:** Capture compound's complexity; too many elements and sites may lead to less stable/metastable phases. Number of sites is considered in magnetic moment and volume per atom computation (see backup slide).



Feature: Bond structure

Captures bond's (min/avg/max) distance and coordination using MinimumDistanceNN [8]. Features are:

- ▶ **Number of bonds:** coordination and connectivity
- ▶ **Unique bond type:** structural diversity and bonding motifs
- ▶ **Bond length mean and standard deviation:** average bonding length and geometrical distortion

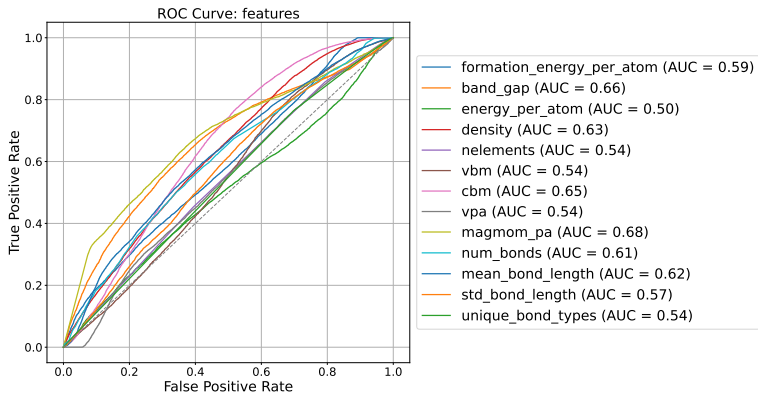


Receiver Operating Characteristic Curve

Receiver Operating Characteristic (ROC) Curve: A graphical tool for evaluating the classification performance of models or features across varying decision thresholds.

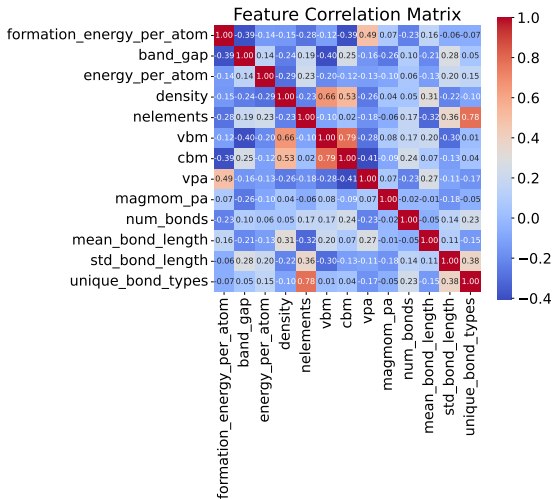
Area Under Curve (AUC): A quantitative summary of overall discriminatory power—higher values indicate better separation between classes.

ROC curves: Selected input features, with respective AUC in the legend. The relatively low AUC scores indicate the features' limited ability to differentiate between stable and unstable chemical compounds.



Correlation Matrix

XGBoost [3] was used to analyze relationships among the ML input features. Only the general features' pairwise correlations are visualized below to avoid confusion.



Machine Learning Methods

- ▶ **Deep Neural Network (DNN):** This work focuses on assessing the effectiveness and potential of DNN in predicting materials properties because of its:
 - ▶ Flexible architecture
 - ▶ Strong capacity for capturing complex, nonlinear relationships in the data
- ▶ To validate and benchmark DNN performance, two additional models were employed:
 - ▶ **Logistic Regression (LR):** A fast, interpretable baseline model for comparison with DNN results.
 - ▶ **Random Forest (RF):** A tree-based ensemble model chosen for its robustness on small datasets and its ability to provide insight into feature importance.

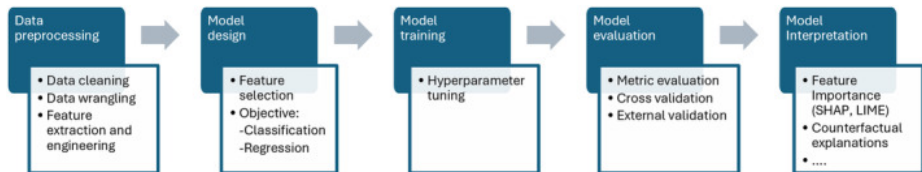


Figure: Reference [9]

Preprocessing Features

Feature Normalization: Help the model train more efficiently and accurately [5]:

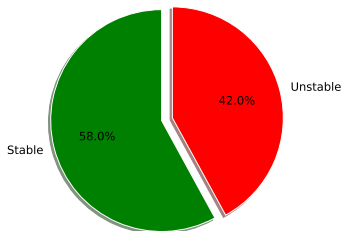
- ▶ Used `StandardScaler` from `scikit-learn`.
- ▶ Plays a crucial role during pre-processing features by ensuring each input feature has $\mu = 0$ and $\sigma = 1$.
- ▶ In **DNN** improves:
 - ▶ Gradient descent functionality as features are on similar scales
 - ▶ Avoid gradient fluctuation in deep layers
 - ▶ Works well with activation functions that are sensitive to input scale like sigmoid
- ▶ Helps stabilize DNN, Logistic Regression, and Random Forest model learning.

Class Weight

The dataset is moderately imbalanced, with a higher proportion of stable/metastable compounds. To mitigate this, class-level weight is used:

- ▶ To address class imbalance, uniform weights are assigned using `class_weight='balanced'` from `scikit-learn`.
- ▶ it automatically adjusts weights inversely proportional to class frequencies, encouraging the model to pay more attention to the minority class.
- ▶ Applied to all three ML methods.

Chemical Compound Stability Distribution (Train Set)



ML Methods Hyperparameter Tuning

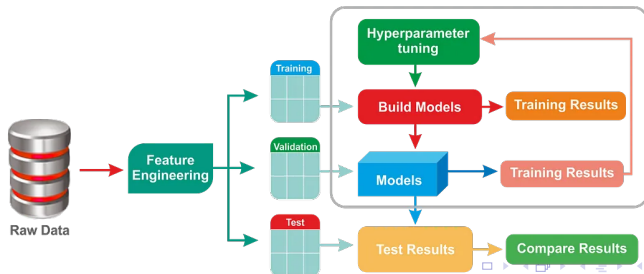
Goal: Find the best combination of ML model parameters that optimizes its performance

Hyperparameters: Settings chosen before training, crucial to maximize model accuracy and effectiveness.

Tuning Method:

- ▶ Use GridSearchCV from scikit-learn
- ▶ Performs exhaustive search over predefined parameter values
- ▶ 3-fold cross-validation ($cv=3$) used to reduce overfitting

Optimization criteria: *Accuracy* (overall correctness), *F1 score* (balance between precision and recall), and *the area under the ROC curve* (model's power to distinguish among classes).

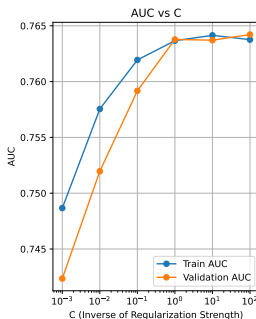
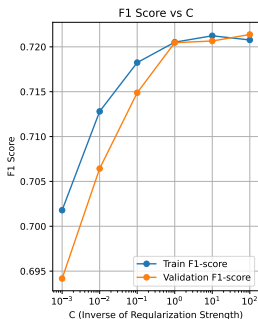
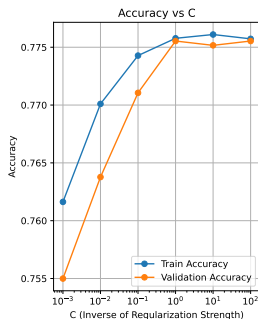


LR Hyperparameters

► LogisticRegression: Hyperparameters used in tuning LR:

Hyperparameter	Values	Description
C	[0.001, 0.01, 0.1, 1 , 10, 100]	Inverse of regularization strength
penalty	l1 , l2	Type regularization used to prevent overfitting
max_iter	[200, 500 , 1000]	Max iterations to converge

(The bold, red values are the hypertuned parameters)

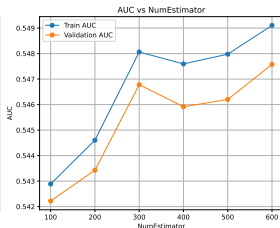
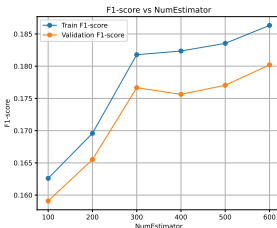
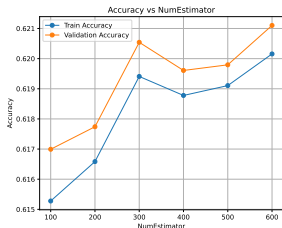


RF Hyper Parameters

► RandomForest: Hyperparameters used in tuning RF:

Hyperparameter	Values	Description
n_estimator	[100, 200 , 300, 400]	Number of decision trees
max_depth	[5 , 10, 20]	Number of splits/decision levels from the root node to the deepest leaf node
min_samplesplit	[2 , 5]	Minimum number of samples required to split an internal node

(The bold, red values are the hypertuned parameters.)



DeepNeuralNet

- ▶ *Tensorflow* is used to build the neural network [1]
- ▶ **Activation Function:** Rectified Linear Unit (ReLU) ($\text{ReLU}(x) = \max(0, x)$) and sigmoid functions.
- ▶ Prevent overfitting during training:
 - ▶ **Random dropout of neurons:** 10% for all the layers
 - ▶ **Batch normalization:** Stabilize learning by normalizing input values per layer;
 - ▶ **Early stopping:** Prevent validation loss increment;
- ▶ **Learning rate:** Optimizes model weights in response to the estimated error (loss)

DNN Parameters

► Fixed:

- **Architectural regularization:** Including batch normalization, dropout, and early stopping
- **Learning rate:** Used with learning rate scheduler to lower the learning rate as the validation loss plateaus
- **Activation Function:** *ReLU* function for input & hidden layers, *sigmoid* function for output layer
- **Epochs:** Number of complete passes of the entire training dataset through a learning algorithm. To avoid overfitting, `EarlyStopping` callback tool from `keras` was used.

► Tune Hyperparameters to optimize DNN performance

- Number of hidden layers
- Batch size
- Number of hidden units (neurons)

► Metrics: *Accuracy, AUC, and F1-Score*. Next two slides shows DNN mean scores after using three cross-validations. The mean scores with hyperparameters for AUC and F1-Scores metrics are shown in backup slides 3.

DNN Hyperparameter Selection Rationale

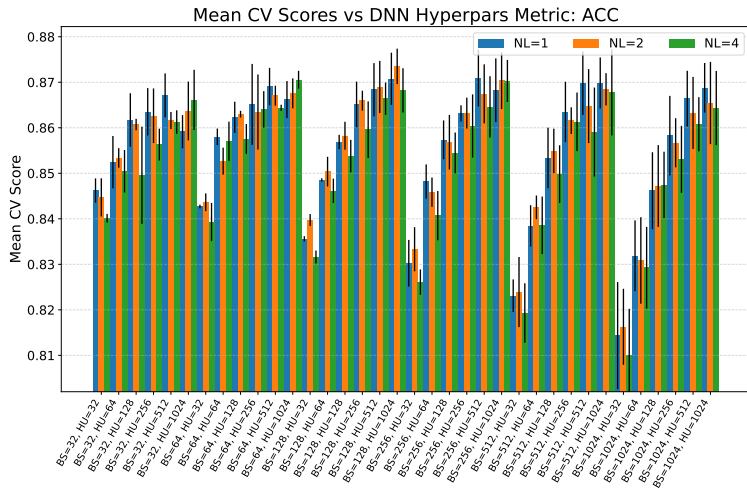
DNN was trained using 108 combinations of hyperparameters via *GridSearch* and *StratifiedKfold* from *sklearn*.

Hyperparameter	Values	Description
Batch_size (BS)	[32,64, 128 ,256,512,1024]	Number of training samples processed at once before updating weights
Hidden_units (HU)	[32,64, 128 ,256,512,1024]	Number of neurons within a layer
Num.HiddenLayers (NL)	[1 ,2,4]	Number of hidden layers

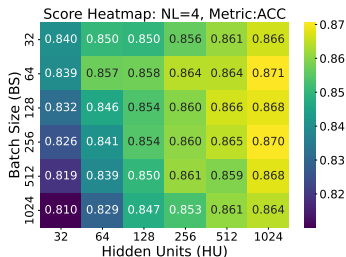
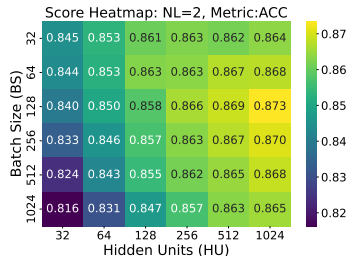
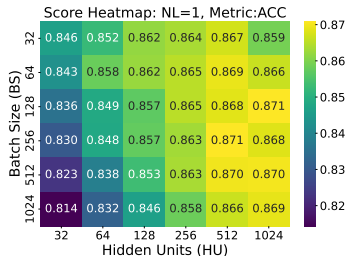
The hyperparameters highlighted in red in the above table were chosen based on:

- ▶ **Optimal Performance:** These values were selected because the validation plots (Slide 1) show that the loss, accuracy, and F1-score metrics have converged, reaching a stable and predictable plateau. This indicates that the model has been sufficiently trained without overfitting.
- ▶ **Performance vs. Efficiency:** Although an alternative hyperparameter combination (BS=128, HU=1024, NL=1) yielded a 2% higher accuracy, the selected set was chosen as the final configuration for this study due to other considerations (e.g., computational efficiency or generalization).

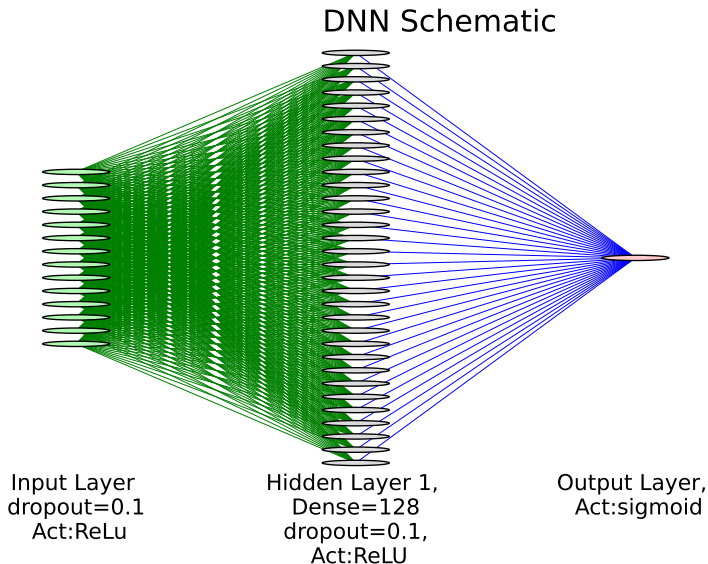
DNN Hyperpars - Barchart - Accuracy



DNN Hyperpars - Heatmap - Accuracy



DNN - Schematic



DNN Model Agnostic Methods

Post-Hoc Analysis:

- ▶ Class of techniques used to evaluate the DNN model as a black box
- ▶ Analyze by observing how DNN's output changes in response to the change in its inputs

Model agnostic methods used in this study:

- ▶ **Partial Dependence Plots (PDPs):** To visualize the relationship between a feature and the model's predictions.
- ▶ **Permutation Feature Importance** Measures the decrease in a model's performance when a single feature's values are randomly shuffled.
- ▶ **SHapley Additive exPlanations (SHAP):** To calculate the contribution of each feature to a specific prediction, used to create both local and global explanations.

DNN - Partial Dependence Plots

Partial dependence Plots (PDP):

- ▶ Visual technique to understand the relationship between DNN prediction and features
- ▶ The effect of the rest of the features is averaged on the model probability
- ▶ Helps understand how model prediction varies with feature values by illustrating where the feature has a stronger or weaker influence.
- ▶ The test datasets are used to make the plot

Caveats:

- ▶ Works best with uncorrelated features
- ▶ As only the average probability is used, the spread in probability is ignored

Implementation: Used partial dependence plots from `scikit-learn` to interpret a deep learning model trained with tuned hyperparameters, wrapped for `scikit-learn` compatibility using `KerasClassifier`.

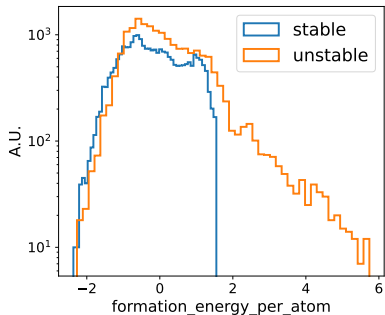
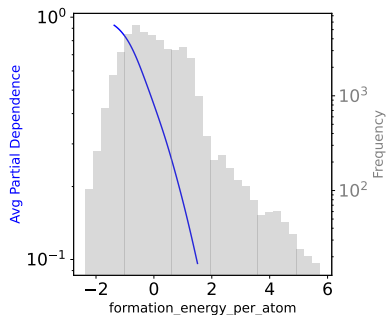
The next two slides show the PDP slides with the feature distribution alongside the stable and unstable feature distribution. The PDP for the rest of the features is shown in the backup slides.

DNN PDP: Energy Feature

Formation Energy per Atom: Energy change when a compound is formed from its constituent elements in their stable energy state:

- ▶ Negative formation energy: more thermodynamically stable compound, less likely to undergo spontaneous decomposition
- ▶ Positive formation energy: less thermodynamically stable, prone to spontaneous decomposition
- ▶ The relationship is nonlinear

N.B: The horizontal axis shows the scaled normalized formation energy values



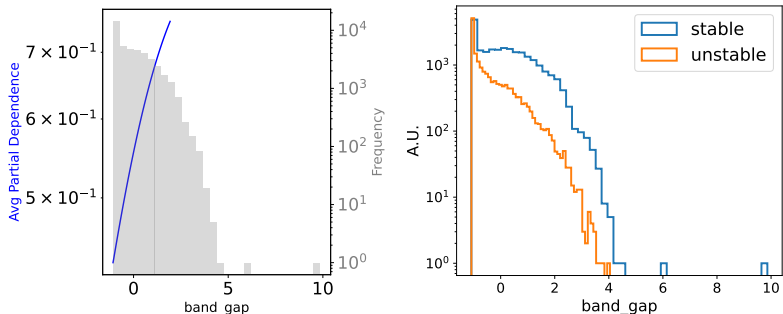
DNN PDP: Electronic Feature

Bandgap: Energy difference between the top of the valence and the bottom of the conduction bands

- ▶ **Large bandgap:** Insulator
- ▶ **Intermediate bandgap:** Semiconductor
- ▶ **Small bandgap:** Conductors

Stable compounds are more semiconductor material than unstable compounds \Rightarrow *PDP increases* in intermediate bandgap values

N.B: The horizontal axis shows the scaled, normalized bandgap values

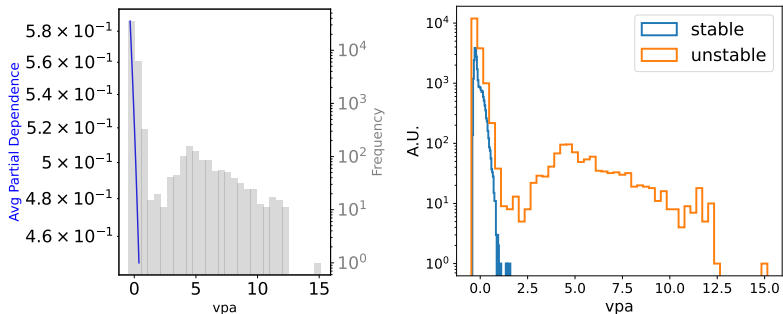


DNN PDP: Structure Feature

Volume per Atom (vpa): Average volume occupied per single atom in the compound

- ▶ Inversely related to atomic density
- ▶ Directly related to the packing efficiency of atoms in a crystal lattice
- ▶ Large vpa: Unstable compound, as highly expanded or porous structures
- ▶ Small vpa: More stable compounds

N.B: The horizontal axis shows the scaled, normalized mean bond length

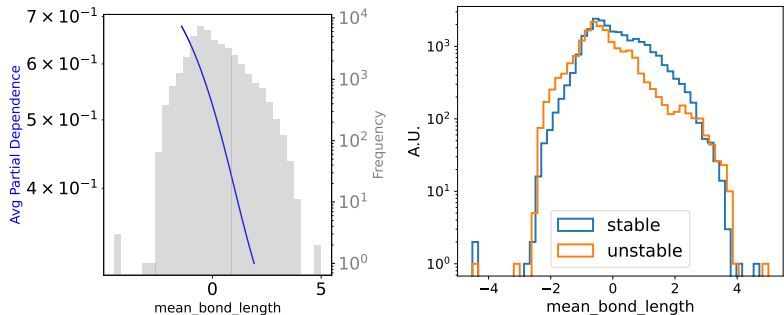


DNN PDP: Bond Structure Feature

Mean Bond Length: Average distance between the nuclei of chemically bonded atoms in a material

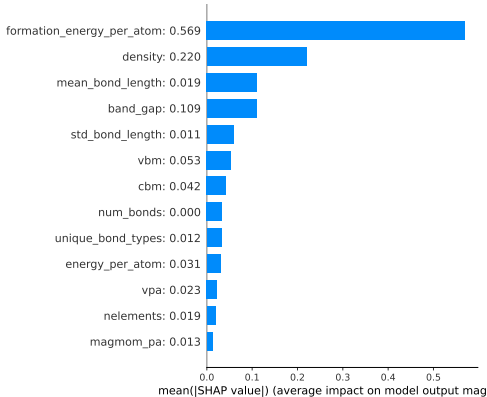
- ▶ Direct indicator of the strength and nature of interatomic bonds
- ▶ Large mean bond length: Larger atoms less Stable compounds
- ▶ Short mean bond length: Higher bond orders \Rightarrow increased electron sharing
- ▶ Optimal range: moderate mean bond length

N.B: The horizontal axis shows the scaled normalized mean bond length; hence, the negative values are datasets with lower-than-average mean bond length values



Permutation Feature Importance

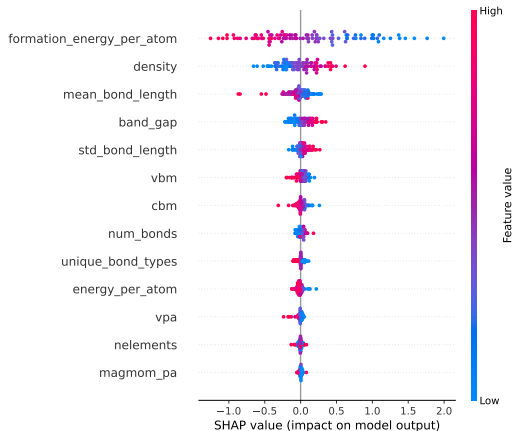
- ▶ **Criteria:** Explains feature contribution to the model accuracy by measuring feature importance and observing how its performance drops when a feature is randomly shuffled. If shuffling a feature makes the model significantly worse, that feature is considered important.
- ▶ **Selected Features:** Features are selected among energy and electronic, structural and composition, and bond features.
- ▶ The test datasets are used to make the plot



SHapley Additive exPlanations - SHAP

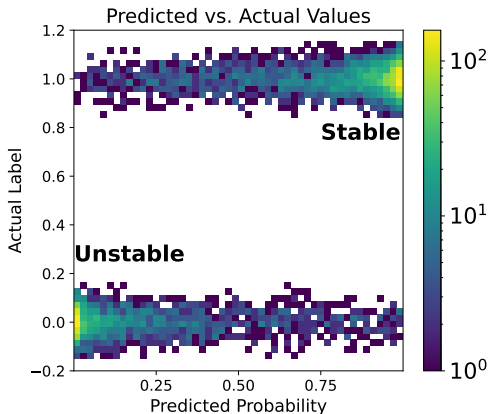
Key points:

- ▶ **Y-axis:** Features from most to least important from the top to bottom
- ▶ **X-axis:** SHAP values, the magnitude and direction of a feature's impact on the model's prediction. Features with a wider spread are more important
- ▶ **Selected Features:** Features are selected among energy, electronic, structural and composition, and bond features.
- ▶ The test datasets are used to make the plot



DNN - Performance

- **Predicted probability:** Test datasets, actual stability is jitter by 5% to help visualization
- **Validation Plots:** Train & Validation datasets (next slide)



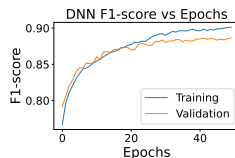
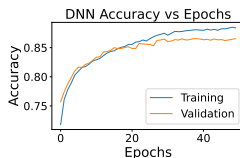
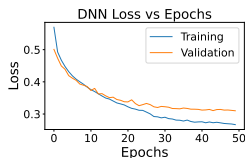
DNN Evaluation: Validation plots

Performance Expectations:

- ▶ **Convergence:** Training and validation metrics plateau as the number of epochs increases \Rightarrow DNN model has converged.
- ▶ **Performance Trend:** Loss should decrease, accuracy and F1-score should increase with more epochs.

Common Observations & Explanations:

- ▶ **Performance Gap:** Between training and validation metrics is expected because the model fits the training data more closely through repeated weight updates.
- ▶ **Impact of Dropout:** validation loss is lower than training loss (and validation accuracy is higher than training accuracy) at the beginning of training due to the dropout regularization applied during training.



DNN Statistical Analysis

► **Core Classification Metrics**

- Accuracy
- Precision
- Recall
- F1-Score
- Specificity
- False Positive Rate

► **Probabilistic and Discriminative Metrics:**

- ROC Curve (Receiver Operating Characteristic)
- AUC (Area Under the Curve)
- Log-Loss (Cross-Entropy Loss)

► **Model Robustness and Generalization**

- Uncertainty Quantification

Core Classification Metrics

Confusion Matrix

Actual	Predicted	
	Unstable	Stable
Unstable	1904 (TN)	273 (FP)
Stable	431 (FN)	2747 (TP)

- ▶ **Accuracy** = $\frac{TP+TN}{TP+FP+TN+FN} = 87.94\%$
- ▶ **Precision** = $\frac{TP}{TP+FP} = 89.7\%$, **Recall** = $\frac{TP}{TP+FN} = 90\%$
- ▶ **F1-Score** = $2 * \frac{Precision * Recall}{Precision + Recall} = 0.898$
- ▶ **Specificity** = $\frac{TN}{TN+FP} = 0.87$, **False Positive Rate** = $\frac{FP}{TN+FP} = 0.13$

Conclusion based on classification metrics:

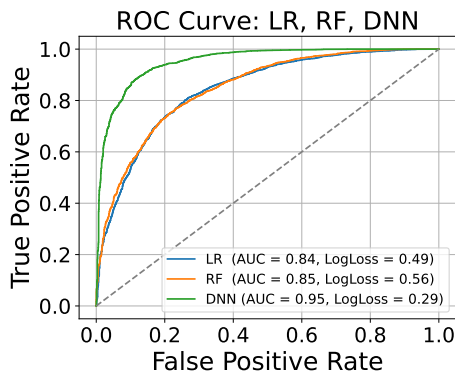
- ▶ high accuracy and a good balance between precision and recall
- ▶ It makes correct positive predictions, and is effective at identifying a majority of the actual positive cases
- ▶ Notable number of false negatives, which should be a focus for future improvement, depending on the application.
- ▶ Low log-loss value confirms the model's high confidence in its predictions.

⇒ **Good Overall Performance**

Probabilistic and Discriminative Metrics

- ▶ Receiver Operating Characteristic (ROC) Curve
- ▶ Area Under the Curve (AUC)
- ▶ Log-Loss: Cross-Entropy Loss

Model	AUC	Log-Loss
LogisticRegression	0.84	0.49
RandomForest	0.85	0.56
DNN	0.95	0.29



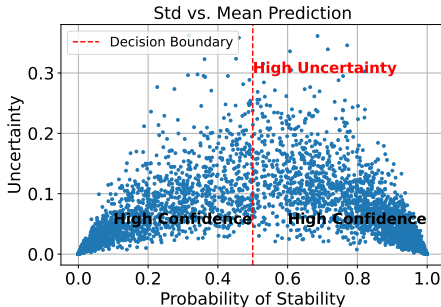
DNN Uncertainty Quantification

Methodology:

- ▶ Training Ensemble of 5 identical DNNs on the same training and validation datasets.
- ▶ Predictive probabilities generated for the test dataset using each of the 5 models
- ▶ Calculate mean (μ) and standard deviation (σ) of the 5 probability predictions for each test dataset to quantify uncertainty

Interpretation:

- ▶ High-confidence areas: high μ and low σ indicates strong model agreement
- ▶ High-uncertainty: high σ around the decision ($\mu=0.5$) boundary indicates low confidence in model disagreement.



Summary and Results

- ▶ **High Performance:** The DNN achieved an accuracy of 88.6%, with a precision of 90.9% and a recall of 89.6% with a discrimination power of 94% with the test datasets.
- ▶ **Robustness:** Systematic hyperparameter optimization was conducted across 108 combinations, leveraging *GridSearch* and *StratifiedKFold* to identify the optimal model configuration.
- ▶ **Superiority:** The DNN model outperformed traditional ML models, demonstrating a 10% increase in discrimination power over both Logistic Regression and Random Forest
- ▶ **Interpretability:** Model sensitivity analysis, using methods like SHAP, confirmed that the most important features aligned with domain expertise
- ▶ **Reproducibility:** The entire project pipeline, including all code and data, is fully documented and available at [7]

Future Direction

- ▶ **Enhance Model Robustness:** Evaluate the DNN model on an out-of-distribution dataset from a new chemical system to rigorously test its robustness and domain transferability.
- ▶ **Predicting Synthesizability:** Expand the project scope to move beyond thermodynamic stability prediction toward the more challenging task of predicting material synthesizability.

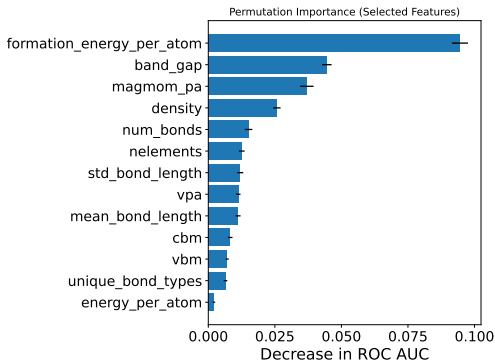
References

- [1] Martín Abadi et al. *TensorFlow: A System for Large-Scale Machine Learning*. USENIX Association, 2016, pp. 265–283.
- [2] Christopher J. Bartel. “Review of computational approaches to predict the thermodynamic stability of inorganic solids”. In: *Journal of Materials Science* 57.3 (2022), pp. 10478–10520. DOI: 10.1007/s10853-021-06865-6.
- [3] Tianqi Chen and Carlos Guestrin. *XGBoost Documentation: Parameters*. <https://xgboost.ai/readthedocs.io/en/latest/parameter.html>. Accessed: [Insert date]. 2023.
- [4] Geoffroy Hautier et al. “Data Mined Ionic Substitutions for the Discovery of New Compounds”. In: *Inorganic Chemistry* 50.2 (2011), pp. 656–663. DOI: 10.1021/ic100504j.
- [5] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015, pp. 448–456.
- [6] Anubhav Jain et al. “Commentary: The Materials Project: A materials genome approach to accelerating materials innovation”. In: *APL Materials* 1.1 (2013), p. 011002. DOI: 10.1063/1.4812323.
- [7] Sara Nabili. *Chem: Machine Learning for Material Stability*. <https://github.com/snabili/chem>. Accessed: 2025-05-27. 2025.
- [8] Hillary Pan et al. “Benchmarking Coordination Number Prediction Algorithms on Inorganic Crystal Structures”. In: *Inorganic Chemistry* 60.3 (2021). PMID: 33417450, pp. 1590–1603. DOI: 10.1021/acs.inorgchem.0c02996. eprint: <https://doi.org/10.1021/acs.inorgchem.0c02996>. URL: <https://doi.org/10.1021/acs.inorgchem.0c02996>.
- [9] Y. Zhou, Y. Wang, Y. Lu, et al. “Thermodynamic Stability Prediction of Inorganic Materials Using Deep Learning”. In: *Computational and Structural Biotechnology Journal* 22 (2024), p. 70056. DOI: 10.1016/j.csbj.2024.70056.

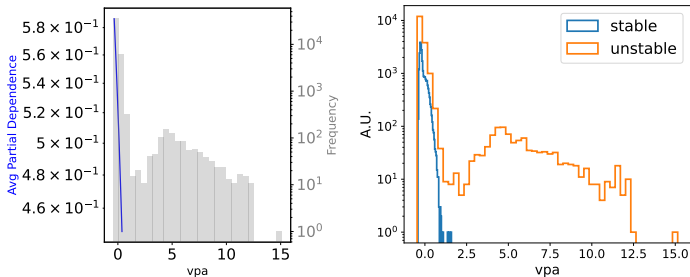
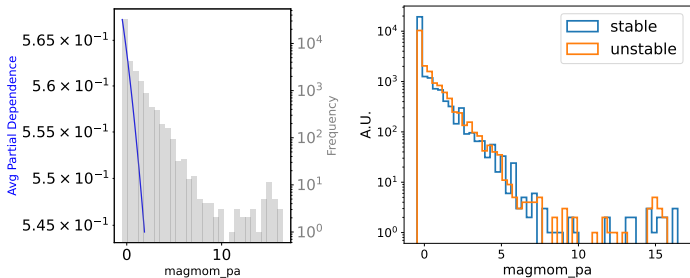
Backup Slides

XGBoost Permutation Feature Importance

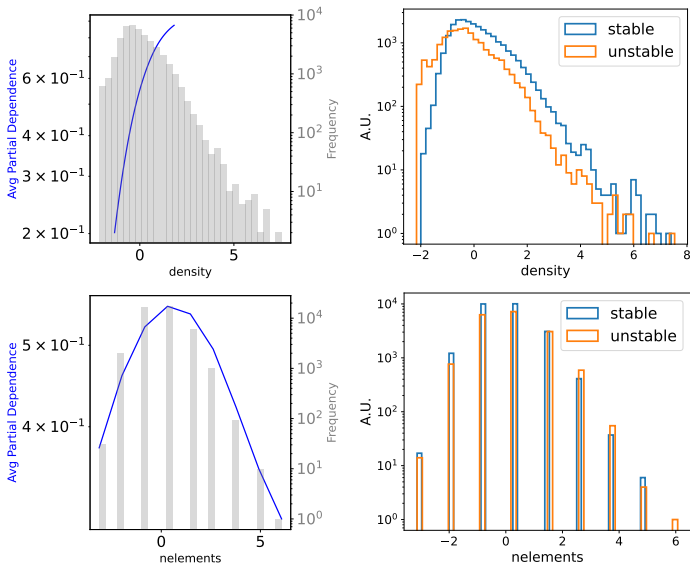
- ▶ Explains the contribution of a feature to the model accuracy. XGBoost built-in feature importance function represents the overall usage of a feature by the tree nodes
- ▶ Used XGBoost to compute, with metric set to area under the ROC curve. Feature importance is assessed 20 times by shuffling its values and checking how the model's performance changes.
- ▶ A handful of selection features are used to make the permutation importance plot, the plot shows the mean value of multiple evaluations and the standard deviation of its evaluation as the error bars.



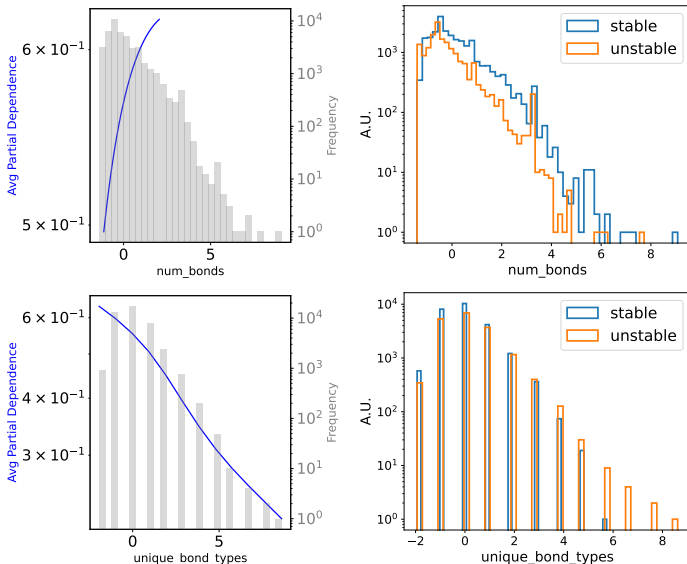
DNN PDP: Structure & Composition Features



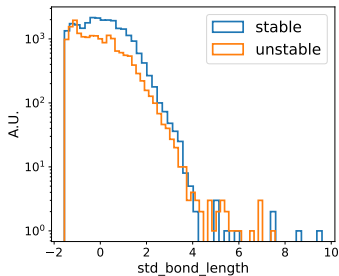
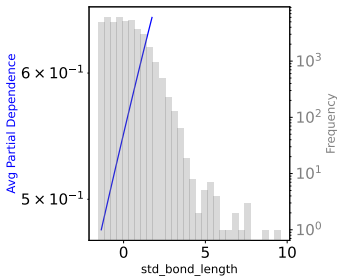
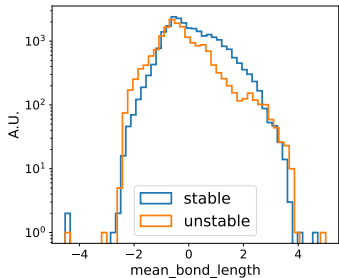
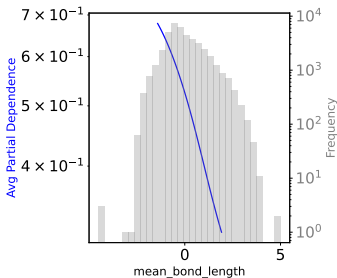
DNN PDP: Structure & Composition Features- Continue



DNN PDP: Bond Structure Features

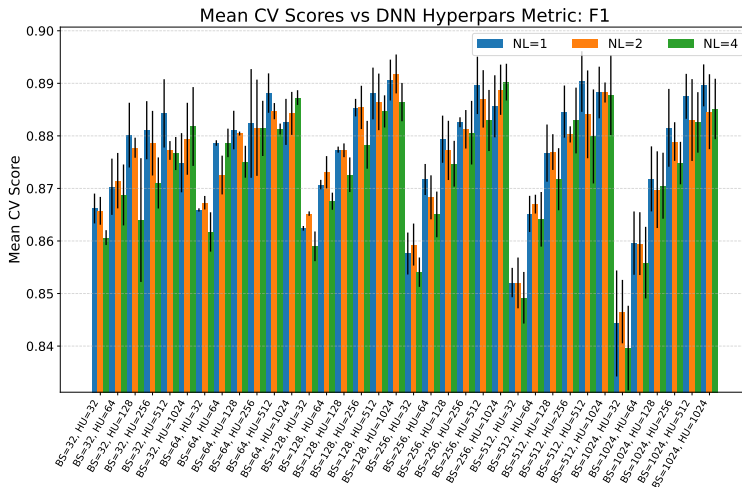


DNN PDP: Bond Structure Features - Continue

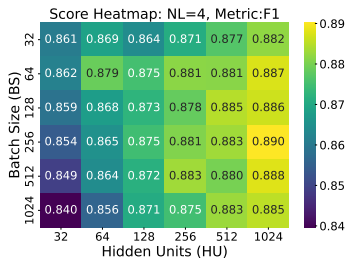
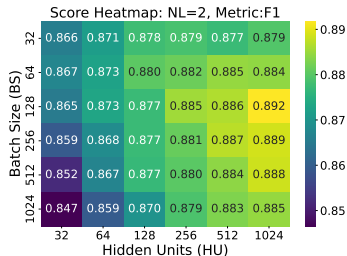
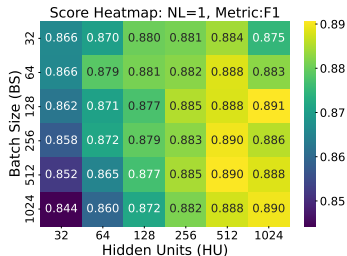


DNN - Hyperpars Tuning - BarChart - F1-Score

Hyperparameter	Values	Description
Batch_size	[32,64, 128 ,256,512]	Number of training samples processed at once before updating weights
Hidden_units	[32,64, 128 ,256,512,1024]	Number of neurons within a layer
Hidden_layers	[1 ,2,4]	Number of hidden layers

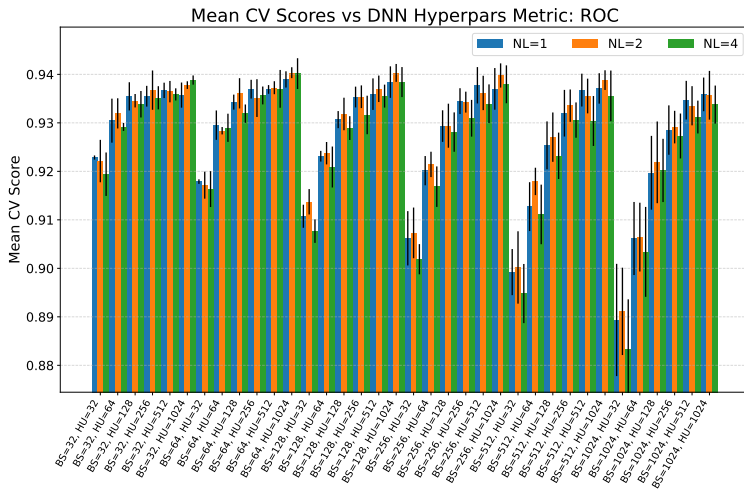


DNN - Hyperpars Tuning - Heatmap - F1-Score



DNN - Hyperpars Tuning- BarChart - AUC

Hyperparameter	Values	Description
Batch_size	[32,64, 128 ,256,512]	Number of training samples processed at once before updating weights
Hidden_units	[32,64, 128 ,256,512,1024]	Number of neurons within a layer
Hidden_layers	[1 ,2,4]	Number of hidden layers



DNN - Hyperpars Tuning - Heatmap - AUC

