Exploring the Interpretability of Deep Learning Based Material Property Prediction Methods

1st Zahra JahediBashiz
Department of Computer Science
University of Southern Maine
Portland, USA
zahra.jahedibashiz@maine.edu

2nd William Richards
Khoury College of Computer Sciences
Northeastern University
Portland, USA
richards.wi@northeastern.edu

3rd Xin Zhang
Department of Computer Science
University of Southern Maine
Portland, USA
xin.zhang@maine.edu

4th James Quinlan
Department of Computer Science
University of Southern Maine
Portland, USA
james.quinlan@maine.edu

5th Yuqi Song
Department of Computer Science
University of Southern Maine
Portland, USA
yuqi.song@maine.edu

***Abstract*—Deep learning has revolutionized material science by providing powerful tools for predicting material properties with high accuracy and efficiency. However, the "black box" nature of deep learning models often hinders their widespread adoption and trust in scientific and industrial applications. Therefore, in this work, we explore the interpretability of deep learning-based material property prediction methods, particularly for the bandgap property, which is a critical property in semiconductors and insulators. We investigate and employ various techniques to enhance model transparency, including feature importance analysis, and assess the impact of different optimizers and hyperparameter configurations. By applying these techniques, we aim to demystify the decision-making processes of deep learning models and provide insights into their predictions. Our findings demonstrate the reliability of deep learning models help identify key factors influencing material properties, and foster confidence in AI-driven material science applications. This work underscores the importance of interpretability in deep learning models for predicting material properties.**

***Index Terms*—Material Property Prediction, Interpretability, Deep Learning, Bandgap Property**

I. Introduction

In the ever-evolving landscape of materials science, discovering novel materials stands as both a formidable challenge and a profoundly meaningful endeavor for the entire science area and our society [1]. For example, the urgent demand for clean energy solutions in light of escalating environmental issues is undeniable. The discovery of materials adept at capturing and storing solar energy, or facilitating clean fuel production, holds the potential to catalyze a transformative shift towards a more sustainable future [2]. An important research topic in the design and discovery of novel materials is predicting material properties, especially some functional materials, such as semiconductors widely used in electronics and optoelectronics, piezoelectric materials generate electric charge in response to mechanical stress [3]. Many scholars are leveraging advanced deep learning (DL) and machine learning (ML) techniques to accelerate the prediction of material properties [4]. For instance, the crystal graph convolutional neural networks (CGCNN) [5] framework was proposed that directly learns material properties from the atomic connections within a crystal. Dunn et al. evaluated supervised machine learning models for predicting the properties of inorganic bulk materials [6]. There are also some studies on specific properties, such as piezoelectric modulus [3], XRD spectrum [7], thermal, mechanical, and optical [8].

Most existing research is focused on exploring the use of large-scale deep learning models for predicting material properties, leveraging their ability to process and learn from datasets with high accuracy. These models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and graph neural networks (GNNs), have shown remarkable success in capturing complex relationships and making precise predictions in material informatics [1]. However, a significant gap remains in evaluating the interpretability of these models. Without interpretability, it is challenging to identify which features or patterns the models rely on, making it difficult to validate their predictions and apply them in critical applications. Therefore, exploring the interpretability of the development and assessment of deep learning models is essential to advance their practical utility in material prediction work.

Inspired by recent advancements in evaluating the interpretability of deep learning models, as highlighted in the works of Li et al. [9] and Stergiou et al. [8], this study employs several methods to enhance the interpretability specifically for the material bandgap property. The bandgap property [10] defines the energy difference between the valence and conduction bands of materials, directly influencing their electrical and optical properties. Accurately predicting the bandgap is essential for developing new electronic and photovoltaic materials.

In our approach, we utilize feature importance analysis to identify which features in the dataset, as indicated by

Magpie, strongly impact prediction of the bandgap feature. Additionally, we explore the effects of different optimizers and combinations of hyperparameters to understand how these selected features influence the performance and interpretability of the models. Enhancing the interpretability of models predicting material property ensures the scientific community can trust and effectively utilize powerful deep learning models in the ongoing exploration and innovation for novel material design and discovery.

II. Related Work

A. ML and DL Based Material Property Prediction

Traditional material properties measurements heavily rely on lots of experiments, which are time-consuming and labor-intensive. While deep learning and machine learning models have substantially transformed material property prediction with accuracy close to *ab-initio* calculations [11], but with computational speeds orders of magnitude faster [12]. These sophisticated computational techniques utilize advanced algorithms to analyze extensive datasets, thereby discerning patterns that significantly enhance predictive accuracy over conventional empirical approaches.

Xie et al. [5] designed a flexible machine learning framework for material property prediction and design knowledge extraction, they also applied this method to the design of new perovskite materials. In [13], the authors integrated atomic orbital interaction features in crystal graph convolutional neural network to learn material properties in a robust way. Transfer learning has been used in [14], Jha et al leveraged existing large Density Functional Theory (DFT) [15] computational datasets together with other smaller DFT-computed datasets to build robust prediction models. These studies have rarely analyzed the interpretability of deep learning models in predicting material properties while interpretability is important to material scholars for designing and discovering new potential materials.

B. Interpretability Analysis of Deep Learning Models

Neural networks excel in classification and prediction tasks but are often seen as black-box function approximators, as discussed by Chakraborty et al. [16] and Li et al. [9] demonstrated that they are difficult to verify due to their complex, black-box nature. Some neural networks are constructed to provide human understandable justifications for their output, allowing insights about the inner workings. We call such models interpretable deep networks. Current research focuses on surveying the past and present state of deep learning interpretability research, discusses the construction of interpretable models, analyzes existing method shortcomings, and suggest future research directions [17].

Many existing models are not interpretable and external analysis is necessary to extract justification for model predictions. Research into the interpretability of deep learning models has theoretical and practical value in fields like medical research, unmanned driving, and information security. In particular, Artificial Intelligence in the form of deep neural networks has proven valuable in clinical applications. In the discipline of medical imaging, Salahuddin et al. [18] discuss the limitations of deep learning in the field and offer guidelines and future directions to enhance AI interpretability in clinical workflows.

III. Methodology

In this section, we introduce details of our proposed methods, including our techniques for feature extraction, data retrieval, feature importance analysis, model training, and evaluation method.

Fig 1 describes the research roadmap for determining the viability of predicting the bandgap property of materials using machine learning and deep learning methods. Our research encompasses several key aspects of model development and analysis. First, we aim to identify the most suitable model for bandgap prediction using Magpie features, which involves comparing various machine learning and deep learning methods. Next, we explore different optimizer combinations to evaluate prediction performance in order to improve model efficiency while retaining accuracy. Finally, we perform a feature importance analysis to determine which Magpie features are most relevant for bandgap prediction. This comprehensive approach not only aims to develop an accurate prediction model but also seeks to understand the underlying factors influencing bandgap properties in materials.

A. Feature Extraction using Magpie

To predict the bandgap of materials based on material features using a Random Forest model, we used the Matminer library [1] (matminer version 0.9.2 and numpy version 1.25.2) for feature extraction and RandomForestRegressor for machine learning by following these steps:

1) Collect a dataset of materials with known bandgap properties and their corresponding structural and compositional information.
2) Utilize Matminer's extensive set of feature extractors to generate relevant descriptors, such as elemental statistics, structural complexity, and electronic properties.
3) Train a Random Forest model on the extracted features, leveraging its ability to handle high-dimensional data and capture non-linear relationships between material characteristics and bandgap property.

B. Data Retrieval

We used machine learning techniques and deep learning models to predict the bandgap of materials based on their features. The data is read from a CSV file containing information about materials. Afterward, the data is split into training and testing sets using the `train_test_split` function from scikit-learn, with 80% of the data allocated for training and 20% for testing. This ensures the model's performance is evaluated on unseen data.

[1] https://hackingmaterials.lbl.gov/matminer/

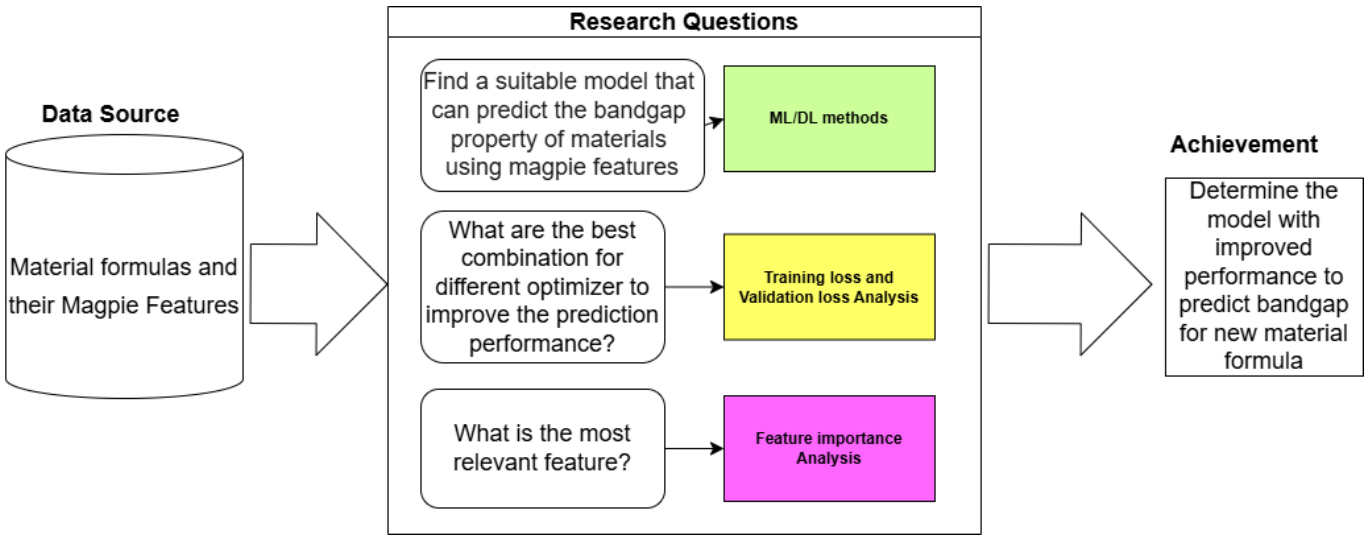

Fig. 1. Roadmap of our work

C. Feature Importance Analysis

There are several methods to detect feature importance in a dataset. One of the most used known interpretable machine learning methods is Random Forest. We fed the data allocated for training is to RandomForestRegressor to find correlations between individual parameters in the data and the bandgap of known materials. The model's performance was evaluated using Mean Squared Error (MSE) and R-squared score on the testing set, providing insight into its accuracy and generalization capabilities. This provides a baseline accuracy against which we can compare our selection of optimizers. We identified and extracted the most influential features contributing to RandomForestRegressor's prediction of bandgap using the the `feature_importances_` function for our trained model and saved the top important features to a CSV file for further analysis and interpretation. We converted the training data to access column names. We retrieved the feature importance to train our model. The top features are identified by sorting the importance in descending order and selecting the indices of the top N features (5, 10, or 20). The corresponding feature names and importances are then saved to a CSV file containing the names and importances of the top features, providing valuable insights into which features have the most significant impact on the model's predictions.

D. Model Training

We selected seven common optimization algorithms (models) at random to evaluate for accuracy on the same training set as our Random Forests baseline. The models selected were: Adam, SGD, RMSprop, AdaGrad, Adadelta, FTRL, and Nadam. These models aim to capture complex patterns and relationships in the data, potentially outperforming traditional machine learning algorithms like Random Forest.

Our experiment constructs neural network models with various architectures, optimizers, and activation functions. Before training we standardize the features using StandardScaler. The training is then performed using the fit function, and the performance is evaluated on a separate validation set.

By changing the number of features in the dataset we train our model on, we can explore different subsets of features and evaluate their relative importance in each model's decision-making process as well as the RandomForestRegressor baseline. The approach allows us to map the importance of specific features to each model's accuracy and aids in interpreting the model under evaluation.

E. Evaluating the Robustness of Proposed Solutions

To demonstrate that our proposed solution can be used to map explanability of optimizers, we evaluate the testing loss and validation loss. By demonstrating that adjusting the number of top features in the training set controls the accuracy of a given model, we show that the importance of particular parameter can be established.

F. A comparison with a white box model

Hyperparameters in DL models for material property prediction, often act as black box measures. These hyperparameters, which may include learning rate, batch size, number of layers, and activation functions, significantly influence model performance but their optimal values are not directly interpretable in terms of material science principles. In contrast, a white box model, such as a simple linear regression or a decision tree, offers more straightforward interpretability. While such white box models are more interpretable, they often lack the predictive power of complex deep learning models, especially for non-linear relationships common in material properties. This trade-off between interpretability and predictive power is a key challenge in material informatics. Our approach of using feature importance analysis helps bridge this gap by providing some level of interpretability to the black box deep learning models. By identifying which Magpie features most significantly impact bandgap predictions, we're adding a layer

of transparency to these complex models, making them more useful for material scientists who need to understand not just what the predictions are, but why they are made.

IV. Experiments

A. Datasets

The dataset used for this work was from the Materials Project database, which is a widely used resource for materials science research. [2]. This database contains information about various materials, including their Material ID, chemical formula, band gap, and numerous features derived from the Magpie featurizer. In this research 'band_gap' column is the target variable for prediction.

B. Evaluation Metrics

MSE Loss Evaluation Since our efforts to use deep learning models to predict `band_gap`(target value) based on material Magpie features is a regression problem, Mean Squared Error (MSE) is the preferred metric over accuracy, as regression problems aim to predict a continuous value, not a classification. This metric weights large errors more heavily than small errors, and lower MSE suggests that the model's predictions are closer to the true values, indicating better performance. The MSE for Random Forests in the test set is 0.58, and the MSE for Adam (our best performing optimizer) is 1.34.

R-squared Prediction Evaluation The R-squared score (`r2_score`) of a model to evaluates the model's influence over the prediction in a range from 0 to 1. The Random Forests regression resulted in a R-squared of 0.76, indicating a strong potential predictability.

C. Performance

Information about the number of layers, learning rate, and batch size used in the experiments. The "Layers" column indicates the depth of the neural network architecture. The "Learning Rate" column specified the rate at which the model adjusted its parameters during training, influencing the convergence speed and optimization performance. The "Batch Size" column defined the number of data samples processed in each iteration of training, affecting the efficiency of gradient descent and the stability of the training process.

1) Different models with hyperparameters: We evaluated the performance of all 7 models in comparison using the following hyperparameter configuration:

- layers = 4
- learning_rate = 0.001
- batch_size = 32

2) Important feature analysis: Table I shows the performance of our seven optimized models across different configurations of neural network architectures. Each row represents a unique combination of optimizer, number of layers, epochs, learning rate, and batch size, along with the corresponding loss and validation loss achieved during training.

[2]https://next-gen.materialsproject.org/materials

TABLE I
Comparison of Optimization Algorithms

Optimizers	# Layers	Epochs	Learning rate	Batch size	Testing_loss	Val_loss
Adam	2	50	0.001	32	1.61	1.58
	4	50	0.001	32	1.44	1.39
	4	100	0.001	32	1.28	1.42
	4	150	0.001	32	2.14	1.89
	4	200	0.001	32	2.27	1.91
SGD	2	50	0.001	32	2.06	1.88
	4	50	0.001	32	2.11	2.01
	4	100	0.001	32	2.05	1.87
	4	150	0.001	32	1.89	1.74
	4	200	0.001	32	1.84	1.71
RMSprop	2	50	0.001	32	2.11	2.03
	4	50	0.001	32	1.77	1.67
	4	100	0.001	32	1.60	1.54
	4	150	0.001	32	1.47	1.66
	4	200	0.001	32	1.53	1.56
AdaGrad	2	50	0.01	32	1.65	1.63
	4	50	0.01	32	1.51	1.44
	4	100	0.01	32	1.33	1.51
	4	150	0.01	32	1.31	1.46
	2	50	0.001	32	23.50	3.82
	4	50	0.001	32	3.62	3.29
	4	100	0.001	32	2.23	2.17
	4	150	0.001	32	2.06	2.07
	4	200	0.001	32	2.60	2.43
Adadelta	2	50	1.0	32	2.46	2.44
	4	50	1.0	32	1.92	1.83
	4	100	1.0	32	1.55	1.61
	4	150	1.0	32	1.53	1.50
	4	200	1.0	32	1.84	1.81
	2	50	0.001	32	1808.77	272.31
	4	50	0.001	32	159.04	7.48
	4	100	0.001	32	7.71	3.84
	4	150	0.001	32	9.19	3.95
	4	200	0.001	32	3.83	3.16
FTRL	2	50	0.01	32	1.84	1.87
	4	50	0.01	32	1.86	2.01
	4	100	0.01	32	1.66	1.84
	4	150	0.01	32	1.54	1.85
	4	150	0.01	64	1.57	1.56
	4	200	0.01	64	1.49	1.65
	2	50	0.001	32	3.33	2.18
	4	50	0.001	32	2.62	2.33
	4	100	0.001	32	2.28	2.27
	4	150	0.001	32	2.21	2.17
	4	200	0.001	64	2.18	2.19
Nadam	2	50	0.001	32	1.86	1.79
	4	50	0.001	32	1.74	1.68
	4	100	0.001	32	1.58	1.80
	4	150	0.001	32	1.47	1.61
	4	200	0.001	32	1.64	1.63

D. Experimental Results

Table I summarizes experimental results for various optimization algorithms applied to a machine learning task, where different subsets of features were used. Each row in the table corresponds to a different configuration of the optimization algorithm and feature set, and the columns represent different parameters and metrics obtained from the experiments.

Algorithm and Feature Set Comparison: Table I compares the performance of our selected optimizers across the top 5, 10, and 20 feature subsets. This allows for a comprehensive evaluation of how the choice of optimization algorithm and feature set impacts the model's performance.

Controlled Algorithm Hyperparameters: Parameters such as the learning rate, batch size, and the number of epochs are held constant across different experiments for each algorithm. This ensures a fair comparison between the algorithms, as they are evaluated under similar conditions.

Performance Metrics: Two performance metrics are reported for each experiment: testing loss and validation loss. These metrics indicate how well the model performed relative to the reference model with a particular configuration of algorithm and feature set performs on the training and val-

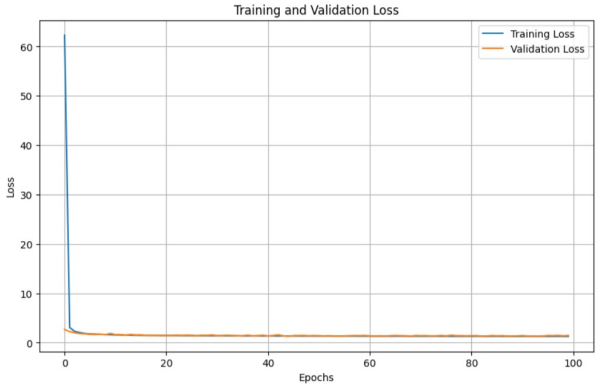

Fig. 2. Training and Validation Loss

idation datasets respectively. Lower loss values indicate better performance because they signify that the model's predictions are close to the actual target values during both training and when evaluated on unseen data. Evaluating the relationship of training and validation loss also helps ensure that the model is not overfitting to the training data, as low training loss coupled with high validation loss may indicate that the model has memorized the training data without generalizing well to new data.

Result Analysis: The plot in Fig 2 shows the relationship between the training and validation loss of the Adam optimizer over 100 epochs. Initial high training loss with a rapid decrease that converges with the validation loss value indicates that the model can generalize its predictions and is not overfitting.

The process is repeated for different combinations of optimizers, activation functions, and model architectures to compare their performance and find the best-performing model.

Finally, we provide insights into how changing the optimizer, learning rate, number of epochs, and model architecture affect the model's performance in terms of predicting the bandgap. By experimenting with different configurations, we can identify the most suitable model architecture and hyperparameters for this specific regression task. Table II displays experimental results from our selected algorithms when trained on the full dataset of 132 features compared to training on only the N most important features. Each row corresponds to a different algorithm, showcasing the performance in terms of training and validation loss over that algorithm's best performing number of epochs. The algorithms are evaluated alongside different sets of features, denoted by `"top_5_features"`, `"top_10_features"`, and `"top_20_features"`. To obtain the experimental results using top features, we selected the following optimization algorithms with their respective epochs:

- Adam with 100 epochs
- SGD with 200 epochs
- RMSprop with 150 epochs
- Adadelta with 200 epochs
- FTRL with 200 epochs
- Nadam with 150 epochs

Table II provides a comprehensive comparison of the algorithms' effectiveness in training the model, highlighting their respective strengths and weaknesses. One notable observation is the performance variation among different optimizers. Adam consistently demonstrates competitive performance across different configurations, achieving relatively low losses compared to other optimizers. On the other hand, Adadelta generally exhibits higher losses across most configurations, suggesting it might not be as effective in optimizing the neural network parameters for the given dataset.

Additionally, the impact of varying the number of layers and epochs is evident. Increasing the number of layers tends to improve model performance, as observed by the decreasing trend in loss and validation loss for most optimizers when transitioning from 2 to 4 layers. Similarly, extending the training duration by increasing the number of epochs often results in further performance improvement, indicated by the decreasing loss values as epochs increase.

TABLE II
PERFORMANCE OF SELECTED MODELS ON REDUCED FEATURE SET

Optimizers	Feature Set	Epochs	Testing_loss	Val_loss
Adam	All-features	100	1.28	1.42
	top_5_features	100	1.42	1.48
	top_10_features	100	1.01	1.02
	top_20_features	100	0.83	0.88
SGD	All-features	200	1.84	1.71
	top_5_features	200	1.52	1.46
	top_10_features	200	1.17	1.12
	top_20_features	200	1.25	1.36
RMSprop	All-features	150	1.47	1.66
	top_5_features	150	1.48	1.16
	top_10_features	150	1.07	1.04
	top_20_features	150	1.04	1.08
AdaGrad	All-features	150	2.06	2.07
	top_5_features	150	1.76	1.63
	top_10_features	150	1.51	1.42
	top_20_features	150	1.42	1.32
Adadelta	All-features	200	3.83	3.16
	top_5_features	200	1.86	1.72
	top_10_features	200	1.67	1.53
	top_20_features	200	1.6	1.46
FTRL	All-features	200	2.18	2.19
	top_5_features	200	1.82	1.71
	top_10_features	200	1.91	1.71
	top_20_features	200	1.38	1.33
Nadam	All-features	150	1.47	1.61
	top_5_features	150	1.4	1.39
	top_10_features	150	0.97	0.98
	top_20_features	150	0.79	0.84

The learning rate shows less direct influence on optimization performance. Optimal learning rates vary depending on the optimizer and other hyperparameters, with higher learning rates sometimes leading to unstable training and higher losses, as seen in some configurations using AdaGrad and FTRL with higher learning rates.

Overall, this comparison highlights the importance of selecting an appropriate optimizer, tuning hyperparameters such as learning rate and network architecture, and monitoring performance metrics like loss and validation loss to effectively train neural network models for a given task.

Observations: Across most experiments, the testing loss is lower than the validation loss, suggesting that the models may be overfitting to the training data.

Some optimizers, such as Nadam and Adam, consistently achieve lower training and validation losses than others like FTRL and Adadelta, indicating some optimizer families are better suited to material prediction tasks.

The size of the feature set used correlates strongly with the performance of each algorithm. The top five features weakly improve performance over the full data set, but the top 10 and 20 features have a stronger correlation. There are instances where the performance improvement from using more features is marginal or even detrimental (e.g., SGD-top_10_features to SGD-top_20_features), indicating potential feature redundancy or noise.

Recommendations: Based on the observed results, it may be beneficial to prioritize algorithms like Nadam and Adam for further exploration due to their consistently strong performance across different feature sets. Feature selection or dimensionality reduction techniques could be employed to identify and retain only the most informative features, potentially improving the overall model performance and reducing overfitting. Fine-tuning hyperparameters such as the learning rate and batch size could further enhance the performance of the algorithms, especially where certain algorithms exhibit sensitivity to these parameters.

Important factors in optimal learning:

1) Choice of optimizer: Optimizers show significant differences in accuracy, independent of the configuration of hyperparameters. For instance, Adam and Nadam consistently show better performance compared to others like Adadelta or FTRL.
2) Number of layers: Increasing the number of layers from 2 to 4 generally improved model performance for most optimizers, as evidenced by decreasing loss values.
3) Number of epochs: Extending the training duration by increasing epochs often led to improved performance, though this wasn't always linear. For example, Adam's performance peaked at 100 epochs and degraded afterwards.
4) Feature set size: The number of features used significantly impacted performance. Interestingly, using the top 10 or 20 features often outperformed using all 132 features, suggesting that feature selection can be crucial for optimal learning.
5) Learning rate: While kept constant for most experiments (0.001), when varied (e.g., for AdaGrad and Adadelta), it showed significant impact on performance. This suggests that learning rate tuning is crucial for optimal performance.
6) Batch size: Although mostly kept constant at 32, the few experiments with larger batch sizes (64) for FTRL showed some performance differences, indicating batch size can affect learning.
7) Model architecture: The combination of layers, epochs, and other hyperparameters constitutes the model architecture, which significantly impacts performance.
8) Overfitting prevention: The relationship between training and validation loss is crucial. Optimal learning occurs when both losses are low and close to each other, indicating good generalization.
9) Task specificity: The optimal configuration appears to be task-specific. For predicting bandgap in materials, certain optimizers (Adam, Nadam) and feature sets (top 10 or 20) consistently performed better.

V. Conclusion

We present an exploration of using machine learning techniques to interpret the importance of data features to optimized models. Our method employs first creating a reference using a Random Forest model with extractable feature importance. It then uses these known important features to train optimizers across an array of hyperparameters and records the accuracy performance of the models on the full data set. Subsequently, the data set is methodically reduced to profile the accuracy of optimizers as the data set reduces to only the most important features.

The experiments demonstrate that this method of comparing performance of a given model to a known set of important features allows us to effectively control the accuracy of the model under evaluation and estimate the influence of the feature set in that model's predictions. Our results indicate that the Adam and Nadam optimizers are well suited to this interpretability method and might inform the development of future models for this task. We are ready to release all the code and benchmark data on GitHub upon the publication of this paper.

In summary, we explore the interpretability of deep learning models for predicting material properties with a particular emphasis on the bandgap property, a crucial determinant in the functionality of semiconductors and insulators. We have employed a variety of interpretative techniques, including feature importance analysis and the assessment of diverse optimizers and hyperparameter configurations, to enhance the transparency and reliability of our models. In the near future, we are going to design an advanced, inherently interpretable model capable of accurately predicting a broader spectrum of material properties.

References

[1] Jianjun Hu, Stanislav Stefanov, Yuqi Song, Sadman Sadeed Omee, Steph-Yves Louis, Edirisuriya MD Siriwardane, Yong Zhao, and Lai Wei. Materialsatlas. org: a materials informatics web app platform for materials discovery and survey of state-of-the-art. *npj Computational Materials*, 8(1):65, 2022.

[2] Daniel P Tabor, Loïc M Roch, Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Joseph H Montoya, Shyam Dwaraknath, Muratahan Aykol, Carlos Ortiz, Hermann Tribukait, et al. Accelerating the discovery of materials for clean energy in the era of smart automation. *Nature reviews materials*, 3(5):5–20, 2018.

[3] Jeffrey Hu and Yuqi Song. Piezoelectric modulus prediction using machine learning and graph neural networks. *Chemical Physics Letters*, 791:139359, 2022.

[4] Siwar Chibani and François-Xavier Coudert. Machine learning approaches for the prediction of materials properties. *Apl Materials*, 8(8), 2020.

[5] Tian Xie and Jeffrey C Grossman. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Physical review letters*, 120(14):145301, 2018.

[6] Alexander Dunn, Qi Wang, Alex Ganose, Daniel Dopp, and Anubhav Jain. Benchmarking materials property prediction methods: the matbench test set and automatminer reference algorithm. *npj Computational Materials*, 6(1):138, 2020.

[7] Rongzhi Dong, Yong Zhao, Yuqi Song, Nihang Fu, Sadman Sadeed Omee, Sourin Dey, Qinyang Li, Lai Wei, and Jianjun Hu. Deepxrd, a deep learning model for predicting xrd spectrum from material composition. *ACS Applied Materials & Interfaces*, 14(35):40102–40115, 2022.

[8] Konstantinos Stergiou, Charis Ntakolia, Paris Varytis, Elias Koumoulos, Patrik Karlsson, and Serafeim Moustakidis. Enhancing property prediction and process optimization in building materials through machine learning: A review. *Computational Materials Science*, 220:112031, 2023.

[9] Xuhong Li, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond. *Knowledge and Information Systems*, 64(12):3197–3234, 2022.

[10] Jason M Crowley, Jamil Tahir-Kheli, and William A Goddard III. Resolution of the band gap prediction problem for materials design. *The journal of physical chemistry letters*, 7(7):1198–1203, 2016.

[11] Stefano Curtarolo, Wahyu Setyawan, Shidong Wang, Junkai Xue, Kesong Yang, Richard H Taylor, Lance J Nelson, Gus LW Hart, Stefano Sanvito, Marco Buongiorno-Nardelli, et al. Aflowlib. org: A distributed materials properties repository from high-throughput ab initio calculations. *Computational Materials Science*, 58:227–235, 2012.

[12] Dezhen Xue, Prasanna V Balachandran, John Hogden, James Theiler, Deqing Xue, and Turab Lookman. Accelerated search for materials with targeted properties by adaptive design. *Nature communications*, 7(1):1–9, 2016.

[13] Mohammadreza Karamad, Rishikesh Magar, Yuting Shi, Samira Siahrostami, Ian D Gates, and Amir Barati Farimani. Orbital graph convolutional neural network for material property prediction. *Physical Review Materials*, 4(9):093801, 2020.

[14] Dipendra Jha, Kamal Choudhary, Francesca Tavazza, Wei-keng Liao, Alok Choudhary, Carelyn Campbell, and Ankit Agrawal. Enhancing materials property prediction by leveraging computational and experimental data using deep transfer learning. *Nature communications*, 10(1):5316, 2019.

[15] Kevin F Garrity, Joseph W Bennett, Karin M Rabe, and David Vanderbilt. Pseudopotentials for high-throughput dft calculations. *Computational Materials Science*, 81:446–452, 2014.

[16] Supriyo Chakraborty, Richard Tomsett, Ramya Raghavendra, Daniel Harborne, Moustafa Alzantot, Federico Cerutti, Mani Srivastava, Alun Preece, Simon Julier, Raghuveer M Rao, et al. Interpretability of deep learning models: A survey of results. In *2017 IEEE smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, Internet of people and smart city innovation (smartworld/SCALCOM/UIC/ATC/CBDcom/IOP/SCI)*, pages 1–6. IEEE, 2017.

[17] Keyang Cheng, Ning Wang, and Maozhen Li. Interpretability of deep learning: A survey. In Hongying Meng, Tao Lei, Maozhen Li, Kenli Li, Ning Xiong, and Lipo Wang, editors, *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 475–486, Cham, 2021. Springer International Publishing.

[18] Zohaib Salahuddin, Henry C. Woodruff, Avishek Chatterjee, and Philippe Lambin. Transparency of deep neural networks for medical image analysis: A review of interpretability methods. *Computers in Biology and Medicine*, 140:105111, 2022.