*Intern Report*

On

**Data-Driven Machine Learning and Deep Learning Approach for Discovery of Stable Spintronic Materials**

By

**Vikash Prasad Soni**

Under the supervision of

**Prof. Pawan Goyal**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR**

**Acknowledgment**

First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Pawan Goyal of the Computer Science and Engineering Department, IIT Kharagpur for allowing me to be a part of this valuable project. I am profoundly thankful for his support, and advice, for giving me intellectual freedom in my work, and for demanding a high quality of work in all my endeavors. His dedication, expertise, and achievements in academic excellence always inspired me when dealing with the intricacies part of this work.

Additionally, I would like to extend my gratitude to my mentors Mr. Sujeet Kumar (Ph.D.) and Mr. Anirudhha Roy (Ph.D.) of the Computer Science and Engineering Department, IIT Kharagpur for providing lots of encouragement during the entire work progress and keep engaging me in new ideas. Their unwavering commitment and enthusiasm, always willing to lend a helping hand during insightful discussions and sharing their expertise are highly encouraged. Lastly, I would also like to extend my appreciation to all the individuals who contributed in some way to the work described in this report.

I feel privileged to have worked alongside such talented individuals, which helped me gain many skills in the subject matter and enrich my academic journey.

Vikash Prasad Soni

**Table of Contents**

**1. Abstract**…..……………………………..….……………………………………………....4

**2. Introduction**………………………….. ..…….……………................................................5

**3. Methods**……………………………….…………………………….……………………...5

3.1 Data Collection………….....................................................................................................5

3.2 Feature Engineering….........................................................................................................7

3.3 ML/DL Classifier…….........................................................................................................7

**4. Result and Discussion**…………………………...……………………...………..………...9

4.1 Random Forest Classifier Predictions..................................................................................9

4.2 Different Models Predictions.............................................................................................10

4.3 DNN Model Predictions………….....................................................................................12

**5. Conclusion**……………………………………...……………..……………………..……19

**6. Future Work**…………..………………………......……………………………………...19

**7. References**…….…………………………………………...……..…...…………………..21

1. **Abstract**

The quest for new magnetic materials is critical for advancing spintronics technology, which has diverse applications in data communication, data storage, and quantum computing. Density Functional Theory (DFT) is a widely used method for material design. Still, its computationally intensive nature hinders its efficiency in exploring the vast chemical design space to identify suitable materials for spintronics.

To address this challenge, Siriwardane et.al. proposed a novel methodology that combines Generative Adversarial Networks (GAN), Machine Learning (ML) classifiers, and DFT for the discovery of novel magnetic material. The CubicGAN model was used to generate new ternary cubic structures, and then ML classifiers were built for screening the ternary magnetic materials. Using DFT calculations these candidates were further validated for stability.

The main focus of the intern was to review this research paper, validate the result, and try several Machine Learning and deep learning models to improve the accuracy of the magnetic classifier part. The stated accuracy (90 %) of the RandomForest Classifier model is validated successfully with slightly different hyperparameters. The accuracy of the provided dataset is also tested with Logistic regression(LR), KNN, Decision Tree (DT), and AdaBoost machine learning algorithms with several base learners. Some hyperparameter tunning methods are also tried to optimize their performance (RandomSearchCV and GridSearchCV). The best result was achieved with the RFC model. The ANN model was developed using Keras and Tensor Flow and the accuracy was stated as (88%) which also is successfully validated by developing the ANN model using Pytorch and the best hyperparameter was selected by Ray-Tunning method along with observing the learning curve for overfitting, underfitting and good fitting conditions.

1. **Introduction**

Spintronic (spin-based electronics) technology has recently emerged by incorporating the spin degree of freedom into conventional charge-based electronics, as spintronic devices offer low power consumption and limited current leakage. Spintronics is considered for many applications like energy harvesting, spin photovoltaics, and data storage [2]. As one of the most important quantum phenomena, the magnetism of materials covers many applications such as high-resolution imaging[3], and spintronic devices [4]**.**

In magnetic materials, the spin-polarized current is readily available due to different populations of spin-up and spin-down electrons [5]. Therefore, magnetic materials are highly favorable for spintronic devices since their spin orientation can be efficiently manipulated using an external magnetic field. Spintronic devices like magnetic field sensors and hard-disk read-heads use giant-magneto resistance (GMR) to control the electron conductivity by aligning the spin direction of two ferromagnetic materials parallel or antiparallel to each other. GMR technology is widely used in the automotive industry, mobile phones, and the medical field. Lately, antiferromagnetic materials have become attractive candidates for magnetic memories due to their benefits over ferromagnetic compounds, like their good stability under external magnetic fields, capability to generate significant magnetotransport effects, and the absence of stray fields. Therefore, both ferromagnetic and antiferromagnetic materials are vital for developing next-generation spintronic applications.

Generally, DFT is used while studying the magnetic behaviour of any material but it is quite computationally expensive, making it less efficient for analyzing large spaces of the material world. Also, DFT is inefficient while analyzing anti-ferromagnetic ground state behavior because it exists in multiple ground states. Therefore to reduce the dependency on the expensive DFT method [6], wehave developed a deep generative model to generate new ternary and quaternary material. This model is then utilized in work [1] to generate ternary cubic materials, from generated materials that are not present in the MP database and are stable go into an ML/DL classifier for a magnetic and non-magnetic classifier for screening the potential spintronic material. The material classified as magnetic further goes for DFT calculation for structural and thermodynamical stability.

Therefore in this period of internship, the aim was to validate the magnetic classifiers and test the available dataset with several ML models and DL models to increase its accuracy.

1. **Methods**

**3.1 Data collection**

The dataset is taken from[1]. Two separate datasets ( atomic dataset, and formula dataset) were available. Both datasets were pulled from the Material Project database (MP dataset) using the pymatgen query [8]. The first dataset contains information on 112 individual elements with 56 elemental and electronic attributes which should be paramount in the context of magnetism of an element or compounds. The attributes of the atomic dataset have been listed in Table 1.

Table 1: The attributes of atomic dataset[1]

| Atomic volume | Is Mendeleev | Column number |
| --- | --- | --- |
| Atomic number | Is nonmetal | BCC energy diff |
| Is meta | Density | First ionization energy |
| Covalent radius | Is metalloid | BCC effective lattice constant |
| Atomic weight | Is alkali | Ground state band gap |
| BCC fermi | BCC magnetic moment | Ground state est BCClatcnt |
| Is d-block | Is f-block | BCC volume pa diff |
| Ground state effluent | Number of unfilled | Number of valence |
| ICSD volume | Oxidation states | Number of s unfilled |
| Polarizability | Number of p valence | Number of p unfilled |
| Number of f valence | Number of f unfilled | Number of d valence |
| Number of d unfilled | Row number | Space group number |
| Ground state volume pa | Number of s valance | Boiling temperature |
| Ground state magnetic moment | BCC volume pa. | Ground state est FCClatcnt |

The formula dataset consists of the formula of 10284 ternary compounds and each formula contains three different elements mapped with magnetic(1) or non-magnetic(0) levels, this dataset is also extracted from the MP database using Pymatgen code.

I tried to extract the data directly from the MP dataset using pymatgen python code[3] so that I could identify the features that have been given in the atomic dataset so that I could understand the strategy or methodology followed in dealing with missing data. Because it would help later on if some data modification (like adding new features) is carried out to increase the model's performance, I was able to extract 26 attributes that matched with the current atomic dataset with slight variation in some features and found that most of the unavailable data are set to zero, also some data which might not be available at that time is available now in MP database and rest of the features I could not identify. I approached the author too for clarification but could not get a response. Therefore, I proceeded directly with the available dataset. One more finding is that the author had considered the elements up to atomic number 112 and not up to atomic number 118. This is because most of the data of attributes mentioned above are not available for these elements.

**3.2 Feature Engineering**

From the available two datasets atomic dataset and formula dataset a new data frame is built which has 112 total features. For each formula present in the formula dataset the weighted average of all 56 features for all the material is being calculated by using the formula given below,

Let the formula is represented as ABC then the weighted average property of the ternary material using the elemental and electronic attributes of individual element is:

avgA B C) / ()

Now the following 56 attributes of ternary material is taken as maximum difference of properties within all elements. Let's say A >B>C is the value order of a property then the new attribute is a difference of maximum and minimum values, for this case AB. Therefore in total there are 112 attributes generated through the feature engineering process. The 1 to 56 attributes are weighted average and the following 56 attributes are maximum difference between the properties.

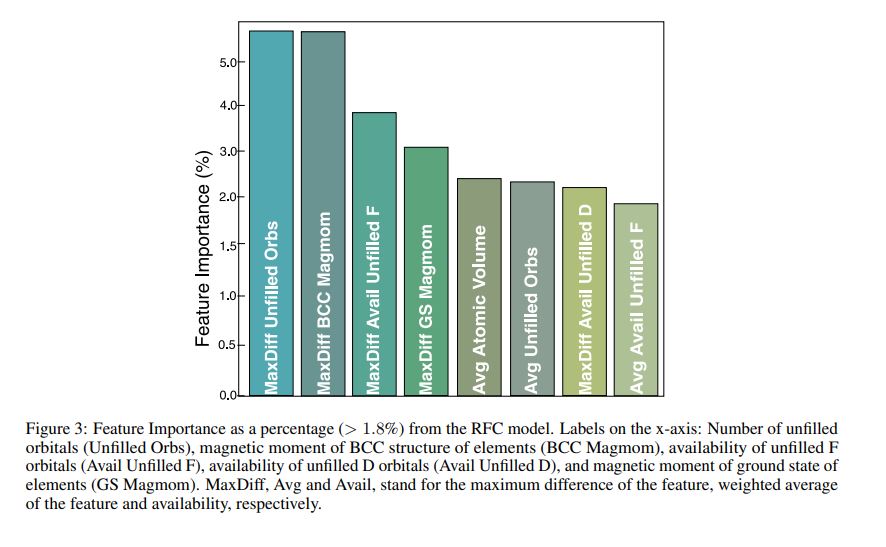
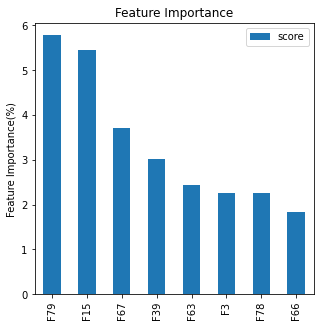
**3.3 Machine Learning Classifier and ANN Classifier**

Before building all the models the final prepared data is split into the inputs and targets dataset using the sklearn library, the dataset is further split into training and testing datasets for machine learning models. First Random Forest Classifier ( RFC ) model is built. The accuracy and the parameters used in the RFC model are compared with the research paper. Then the dataset is also tried with several models( Logistic Regression, KNN, Decision Tree, and Adaboost with different-different base learners ). All the models are also tuned using the RandomSearchCV technique followed by the GridSearchCV technique. RandomSearchCV is used before GridSearchCV to minimize the search space for a grid search. Accuracy and all parameter details of machine learning classifiers are listed and discussed in the result and discussion section.

For ANN model first input dataset is standardized so that generalization of data becomes more accessible and ANN can learn the pattern more accurately and gives good model performance and the data is split into three parts training dataset, validation dataset, and test dataset so that overfitting, underfitting and good fitting of data can be tracked during the tuning the parameters. In this work ANN model is built using Pytorch and using parameters stated in reviewed paper is validated and then several experiment is performed to increase the accuracy of the model, discussed details in the result and discussion section. In experiment-I, the pattern of accuracy and learning rate on several split ratios is observed. In experiment II, The model parameters is tried to tune using the Ray-Tunning method, and the best combination of the parameter from 1000 different combinations of parameters is selected and its performance on different epochs is observed followed by experiment III in which 3 hidden layer with 3 dropout layers are inserted and in experiment-Ⅳ, ANN model with 4 hidden layers is built and in both the experiments parameters are tunned using Ray-Tunning method and performance of best model from 500 samples of parameters is observed at different epochs.

1. **Results and Discussion**

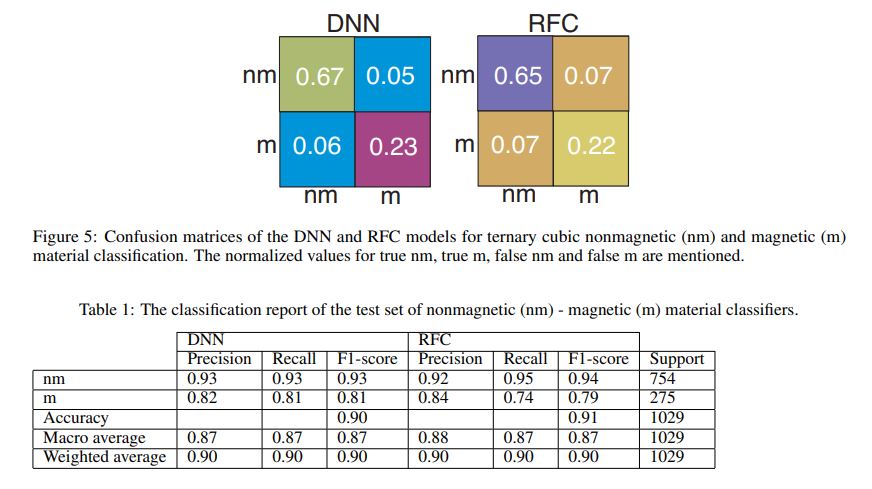
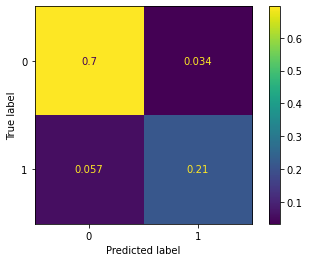
**4.1 Random Forest Classifier Predictions**

**Fig1.**  Feature importance > 1.8 % left (reprinted[1]) , feature importance>1.8 % for validation (right) , F79 , F15 , F67 , F39 , F63 , F3 , F78 , F66 are crossponding features mention in left figure.

The RFC model is a machine learning model which is basically an ensemble technique. It has multiple decision trees in parallel. All decision trees are trained with some defined subsample of the dataset to give output and finally, the majority count in voting is the final output. The mean accuracy of 0.90 0.01 having parameters as maximum depth, decision trees or estimators, minimum samples split, minimum samples leaf, criterion, and max\_features are 80, 1000, 2, 1, Gini, and sqrt respectively. Whereas the accuracy of 0.90 0.01 is also successfully validated in this work. This accuracy is achieved after tuning the parameters using RandomSearchCV followed by GridSearchCV and the final parameters obtained in the current work as maximum depth, decision trees or estimators, minimum samples split, minimum samples leaf, criterion, and max\_features are 100, 870, 2, 1, Gini and sqrt respectively. The Confusion Matrix in normalized form is also compared in Fig1 and the classification report is presented in table 2. In the current work, it can be observed from the normalized confusion matrix that 70% (65% in the reviewed paper ) of the materials are truly classified as non-magnetic and 21 % ( 22% in the reviewed paper ) are predicted as magnetic, The false magnetic and false non-magnetic is less the 5.7% ( 8% in reviewed paper ). The test data set had 74 % (72%) non-magnetic and 26 % (28%) magnetic materials therefore the result are in good agreement in both the rfcs. The accuracy reported as 91% in the classification report is accurately matched but the support values are different in both reports as a test size of 0.1 is considered in the original paper while a test size of 0.2 is considered in the current work. Except that all reports are in good agreement with each other.

**RFC-reprinted**  **RFC - validation**

**Fig2:**  Confusion matrices RFC-reprinted is from a review paper[1] and RFC-validation is a regenerated confusion matrix.

Table 2: Classification report of RF classifier for magnetic and non-magnetic screen RFC-reprinted[1], RFC-validation is from current work.

| RFC-reprinted | | | | | RFC-validation | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | support | Precision | Recall | F1-score | Support |
| nm | 0.92 | 0.95 | 0.94 | 754 | 0.92 | 0.95 | 0.94 | 1500 |
| m | 0.84 | 0.74 | 0.79 | 275 | 0.86 | 0.97 | 0.82 | 557 |
| Accuracy |  |  | **0.91** | 1029 |  | **0.91** |  | 2057 |
| Macro average | 0.88 | 0.87 | 0.87 | 1029 | 0.87 | 0.88 |  | 2057 |
| Weighted Average | 0.90 | 0.90 | 0.90 | 1029 | 0.91 | 0.91 |  | 2057 |

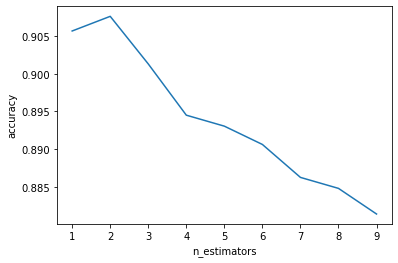
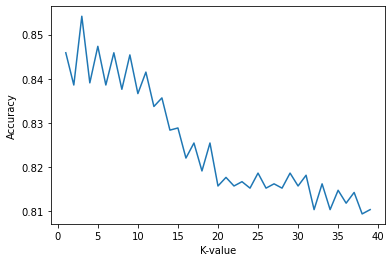
**4.2 Different Models Predictions**

The dataset is also tried on several models (Logistic Regression(LR), KNN, Decision Tree(DT), AdaBoost with base learners as DT, RFC and KNN ). The model performance of all the ML classifiers all listed in Table 3. The parameters of the decision tree are obtained after tuning the parameters using the GridSearchCV technique. DT parameters criterion, max\_depth, min\_samples\_leaf and min\_samples\_split are gini, 27, 3 and 2 respectively. Standarization of the dataset is observed to be very impactful in LR and KNN models the accuracy increased by approx in LR and KNN after standarization of the dataset.

The optimum value of neighbors that k value is 3 for the KNN model is obtained using the elbow method where the accuracy of 0.854 is found to be the maximum for this model. The variation of accuracy and k value for the KNN classifier is shown in fig3(left). AdaBoost classifier with several base models is tested and tuned for better performance. AdaBoost is also an ensemble technique like RandomForest but here the multiple models are arranged in series, the 1st model in the sequence is generally a weak learner whose performance gets boosted by using the next model in the series. The next model in the series focuses more on wrongly predicted values and increases the weight of that particular example and decreases the weight of truly predicted examples. Finally, the majority vote from all models is taken as the final prediction, hence combining all models becomes a strong model. AdaBoost-1 in the listed table consists of DT as a base or weak learner which after tunning gives its best performance as 0.8906 with optimized parameter algorithm, learning\_rate and n\_estimators are SAMME-R, 1.044 and 470 respectively. Though AdaBoost is generally practiced with the weak learner as the base learner also one trial is also given with the RFC model which is generated with an accuracy of 91% as the base learner of AdaBoost and its impact is also observed in the hope that it may work in boosting the accuracy of RFC further but with 50 such estimators the accuracy achieved is 0.88. The impact of the number of RFC in AdaBoost is also observed and it is found that as the number of such estimators increases the accuracy decreases at a learning rate of 1.57 also shown in fig3(right). After hyperparameter tunning AdaBoost-2 model has 0.907 accuracy with parameter base\_estimator, number of estimator and learning rate as RFC model with 0.91 accuracy, 2 and 1.57 respectively.

Table 3: Performance Measurement using ML Classifiers (based on ratio of 80:20)

| Classifier | Precision | | Recall | | F1-score | | Accuracy | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | nm | m | nm | m | nm | m |  | |
| RFC-(paper) | 0.92 0.84 | | 0.95 0.74 | | 0.94 0.79 | | 0.91 | |
| RFC | 0.92 0.86 | | 0.95 0.97 | | 0.94 0.82 | | 0.91 | |
| LR | 0.94 0.60 | | 0.86 0.78 | | 0.90 0.68 | | 0.84 | |
| KNN | 0.93 0.66 | | 0.88 0.77 | | 0.90 0.71 | | 0.85 | |
| DT | 0.92 0.82 | | 0.94 0.78 | | 0.93 0.80 | | 0.89 | |
| AdaBoost-1  (base-DT) | 0.92 0.74 | | 0.91 0.83 | | 0.93 0.79 | | 0.89 | |
| AdaBoost-2  (base - RFC) | 0.96 0.78 | | 0.92 0.87 | | 0.94 0.82 | | 0.91 | |

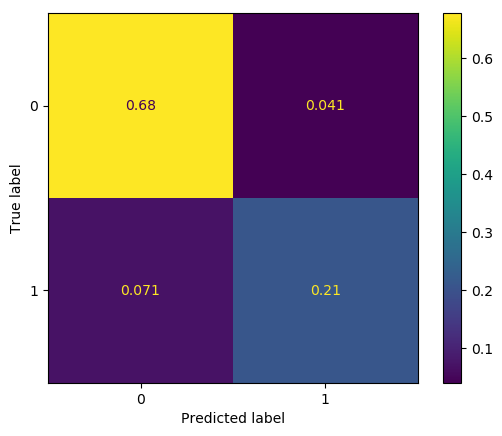
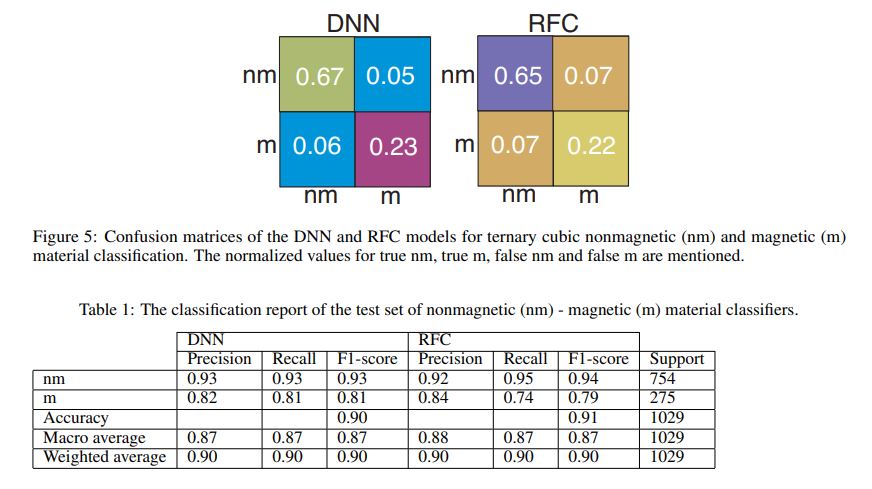


**Fig3:**  Variation of accuracy vs K- value in KNN model (left) , Variation of accuracy in Adaboost model RFC as the base model ( AdaBoost-2) with the number of such estimators at learning rate 1.57

**4.3 DNN Model Predictions**

The dataset is also trained in an ANN model, the ANN model in an original research paper is built using Keras and Tensor Flow while in this work it has been build using Pytorch. For validation purposes the architecture of ANN and hyperparameters are kept the same that is 1 input layer with 112 neurons, 2 hidden layers having 56, and 28 neurons respectively followed by one output layer having one neuron as binary classification. The two two dropout layers with a p-value of 0.5 and L2 regularization were observed to be fruitful for protecting against overfitting. The cost function and optimizer are BinaryCross Entropy and Adams Optimizer respectively. The hyperparameters such as learning rate, batch size and weight decay that is momentum factor are kept as 0.0001, 1500 and 0.001 respectively. With these parameters, the model is trained for 500 epochs. Also, data splitting is kept the same as the 80:20 ratio i.e. 80% of data is used for training, 10% for validation and 10% for testing. The mean accuracy reported as 0.88 in the paper is validated with the current work having an accuracy of 0.888. The validation report is compared with the help of the confusion matrix fig4 and the classification report listed in Table 4. The indeed predicted magnetic material is 0.21(0.23 in reviewed paper). The learning curve of training and validation loss is also presented in the current work to check for the overfitting, underfitting and good-fitting scenarios in fig5. It was observed that with these parameters model was prone to overfit without having the dropout layer and regularization ( Fig5, left ) therefore adding the dropout layer and L2 regularization protected the model from overfit which can be observed from (fig5, right) which is in good generalization of the pattern of the dataset.

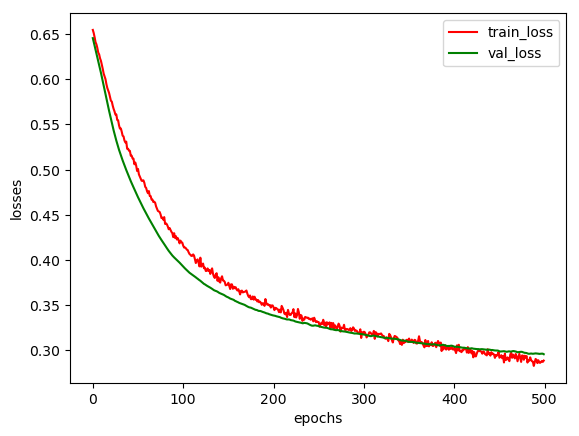
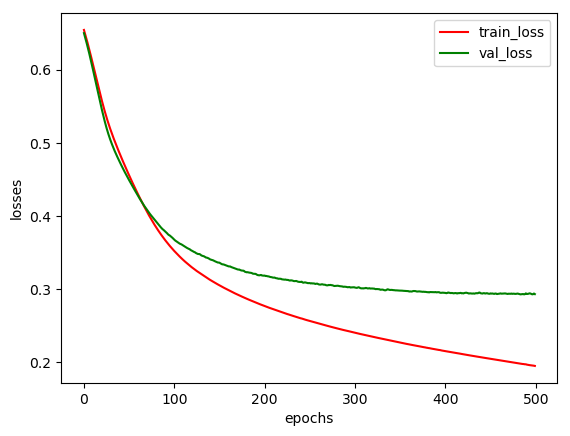
**DNN - Reprinted**  **DNN - Validation**



**Fig4:** left - Confusion matrix DNN-reprinted is from review paper [1] right - Confusion matrix DNN-validation is regenerated confusion matrix ( 0 as non-magnetic, 1 as magnetic )

Table 4: Classification report of DNN classifier for magnetic and non-magnetic screen DNN-reprinted [1], DNN-validation is from current work.

| DNN-reprinted | | | | DNN-validation | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Precision | Recall | F1-score |
| nm | 0.93 | 0.93 | 0.93 | 0.91 | 0.94 | 0.92 |
| m | 0.82 | 0.81 | 0.81 | 0.84 | 0.75 | 0.79 |
| Accuracy |  |  | **0.88** |  | **0.89** |  |
| Macro average | 0.87 | 0.87 | 0.87 | 0.87 | 0.85 | 0.86 |
| Weighted Average | 0.90 | 0.90 | 0.90 | 0.89 | 0.89 | 0.89 |



**Fig5:** Learning curve training and validation loss (ANN model without dropout layer, regularization - overfit at 100 epochs (left), ANN model with 2 dropout layers with p = 0.5 and with L2 regularization - good fit (right)).

Several experiments have been performed to observe the pattern of accuracy on hyperparameters.

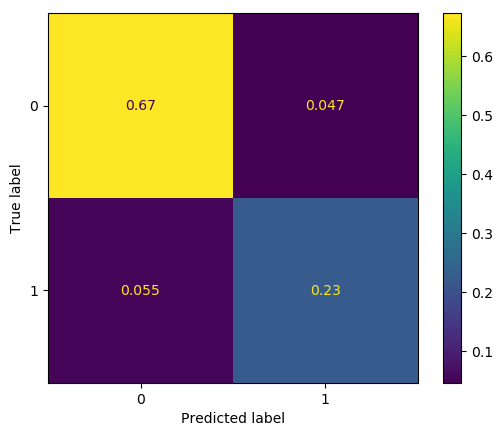
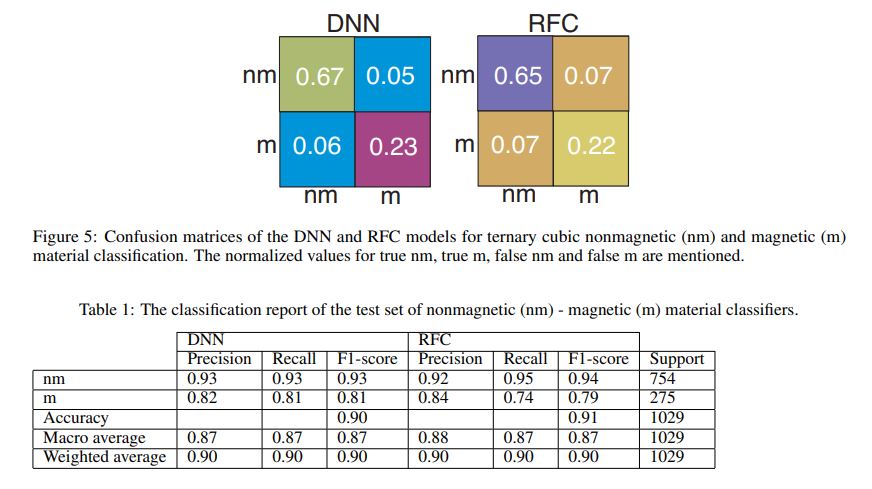
**Experiment-I:**

Keeping the architecture the same i.e with (112,56,28,1) as neural network neurons along with dropouts layers, (splitting of data into 90:5:5, 80:10:10 and 70:15:15 as training:validation: testing) observing the accuracy at different learning rates, epochs. Table 5 illustrates how the learning rate, model accuracy, and number of epochs are related to one another depending on the splitting ratio. It has been observed that a learning rate of 0.01 at 600 epochs with 80:10:10 provided the best result in these trials, with an accuracy of 90% this combination was also observed to have the best confusion matrix result and classification report presented in fig6. The other two splittings i.e. (90:5:5 and 70:15:15) are observed not to produce a good confusion matrix report. From fig6 it can be observed that truly magnetic prediction is 23% which exactly matches with reviewed paper and false predictions are less than 5.5% ( 6% in reviewed paper). This combination was also observed to have a good fit shown in Fig7.

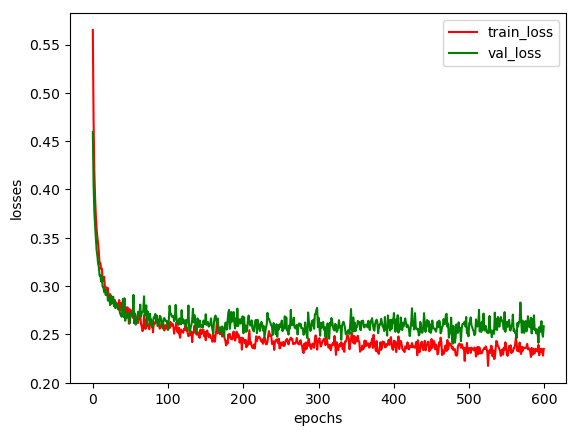
**Experiment-II:**

Using the **Ray Tunning method** for tuning the hyperparameters of the neural network available in Pytorch has been applied to finding number the number of neurons in each hidden layer keeping input neurons as numbers of features used(112), output layer as 1 and split ratio as 80:10:10. Also the tuner has been given a wide range of search space for the number of neurons in hidden layers, learning rate, weight decay and batch sizes for maximum epochs to be 600 and total 1000 samples are collected. Ray Tune method has been implemented with a checkpoint function whose work is to check whether the accuracy is increasing with epochs if not then it automatically stops ( i.e. early stopping ) the further iteration which basically saves time in finding optimum combinations. The tunning is done parallelly in 48 CPUs and 1 GPU.

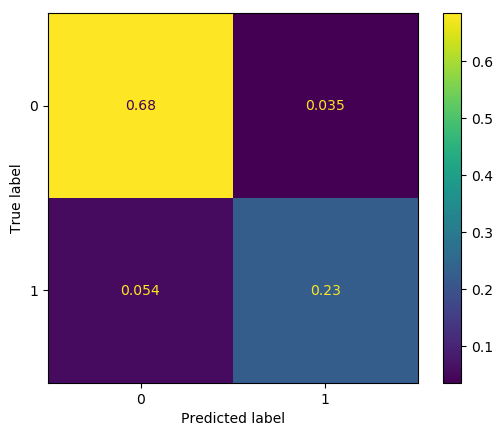
**DNN - Reprinted**  **DNN - validation**



**Fig6:** Confusion matrix DNN-Reprinted (left)[1], Confusion matrix DNN-validation with 80:10:10 split , learning rate as 0.01 with 600 epoch ( accuracy = 90%) (right)



**Fig7:** Loss learning curve for with 80:10:10 split, learning rate as 0.01 with 600 epoch ( accuracy = 90%)



**Fig 8:** Confusion matrix of the best model from ray tunning at 500 epochs with two hidden layers

From ray tunning the best result produced are as follows input layer, hidden1, hidden2, output layer, learning rate, batch size and weight decay as 112, 85, 29, 0.0027, 729 and 0.001 respectively. Using these best parameters produced from ray tunning the model was trained with training data at different different epochs to observe its accuracy based on epochs. At 500 epochs accuracy of 91% is observed (presented in Table 6 ). The classification report and confusion matrix are also presented in Table 7 and Fig 8 respectively.

Table 5: Trials to observe Dependency of learning rate epochs on split ratio

| **Weight\_decay = 0.001**  **Batch\_size = 1500** | | **90:5:5** | **80:10:10** | **70:15:15** |
| --- | --- | --- | --- | --- |
| Learning Rate | Epochs | Accuracy (%) | Accuracy(%) | Accuracy(%) |
| 0.1 | 100 | 77.0 | 82.0 | 82.0 |
| 300 | 74.0 | 79.0 | 83.0 |
| 500 | 78.0 | 81.0 | 82.0 |
| 600 | 83.0 | 80.0 | 82.0 |
| 700 | 78.0 | 82.0 | 85.0 |
| 0.01 | 100 | 88.1 | 88.7 | 87.2 |
| 300 | 87.1 | 89.0 | 88.8 |
| 500 | 88.3 | 89.0 | 89.0 |
| 600 | 89.5 | **90.0** | 89.3 |
| 700 | 88.3 | 89.0 | 89.3 |
| 0.001 | 100 | 88.5 | 89.0 | 88.5 |
| 300 | 88.5 | 89.3 | 89.5 |
| 500 | 87.9 | 89.5 | 89.5 |
| 600 | 88.5 | 89.6 | 89.4 |
| 700 | 88.9 | 89.3 | 89.3 |
| 0.0001 | 100 | 81.9 | 79.6 | 78.6 |
| 300 | 87.3 | 87.0 | 85.5 |
| 500 | 88.5 | 88.0 | 87.4 |
| 600 | 88.5 | 88.2 | 88.0 |
| 700 | 88.7 | 88.4 | 88.2 |

Table 6: Trials on best parameter produced by Ray-Tunning method to observe dependency of epochs and accuracy

| LR = 0.0027  Weight decay = 0.001  Batch size = 729  Neurons = ( 112,85,29,1)  Split ratio = 80:10:10 | Epochs | Accuracy (%) |
| --- | --- | --- |
| 10 | 86.2 |
| 20 | 87.8 |
| 30 | 88.3 |
| 40 | 88.6 |
| 50 | 89.1 |
| 100 | 89.7 |
| 200 | 89.6 |
| 300 | 89.8 |
| 500 | **91** |
| 600 | 89.7 |
| 700 | 89.5 |

Table 7: Classification report of model with best ray tuned parameters at 500 epochs

|  | Precision | Recall | F1-score | Support |
| --- | --- | --- | --- | --- |
| nm | 0.93 | 0.95 | 0.94 | 739 |
| m | 0.87 | 0.81 | 0.84 | 289 |
| Accuracy |  | **0.91** |  | 1028 |
| Macro average | 0.90 | 0.88 | 0.89 | 1028 |
| Weighted Average | 0.91 | 0.91 | 0.91 | 1028 |

From confusion matrix in fig 6 it can be observed that truly predicted magnetic achieved is 23%(23% in reviewed paper) and false positive and false negative predictions are less than 5.4%( 6% in reviewed paper). Slightly better performance that is with 91%(88% in reviewed paper ) is achieved in the modified ANN classifier model.

**Experiment-III : ( 3 hidden layers )**

In this experiment one more hidden layer along with one additional dropout layer is added in the ANN architecture keeping the input layer with 112 neurons and output layer with 1 neuron. The split ratio is 80:10:10 and the model is tunned with the Ray tunning method in pytorch keeping maximum epochs as 600. A total of 500 combinations of hyperparameters (number of neurons in all 3 hidden layers, learning rate, weight decay and batch size ) within a wide range of search space has been observed. The best combination from all 500 combinations of hyperparameters is observed as hidden1, hidden2, hidden3, learning rate, weight decay and batch size to be 96, 88, 41, 0.003, 0.001 and 720 with accuracy as 89.5. Using these parameters specifications model accuracy with 3 hidden layers is observed at different epochs ( table 8). The best result for this experiment was observed at 600 epochs with 21% truly magnetic prediction. Though 90% accuracy was observed at 600 and 700 epochs but truly magnetic predictions were decreased to 20% and the model was very much prone to overfit and increasing the weight decay protects overfit but decreases the accuracy.

Table 7: Classification report of ANN model having 3 hidden layers with best parameters estimated by Ray-Tune is observed at different epochs.

| LR = 0.003  Weight decay = 0.001  Batch size = 720  Neurons = ( 112,96,88,41,1)  Split ratio = 80:10:10 | Epochs | Accuracy (%) |
| --- | --- | --- |
| 10 | 87.7 |
| 20 | 87.8 |
| 30 | 88.8 |
| 40 | 88.7 |
| 50 | 89.2 |
| 100 | 89.3 |
| 200 | 89.3 |
| 300 | 89.7 |
| 500 | 89.8 |
| 600 | 90 |
| 700 | 90 |

**Experiment-Ⅳ : ( 4 hidden layers )**

Similar to experiment-III, the ANN model is built having 4 hidden layers and Using the Ray-Tunning method best parameters were observed from the best of 500 different combinations of hyperparameters. It has been observed that neurons with input, hidden1, hidden2, hidden3, hidden4, output, learning rate, weight decay and batch size to be 112, 105, 68, 36, 102, 1, 0.0023, 0.001 and 720 respectively produced an accuracy of 90% at 100 epochs.

1. **Conclusion**

In material science generally DFT is used to screen based on any property of the material for discovering new materials. Finding new magnetic materials is quite computationally expensive using DFT due to the existence of multiple magnetic ground state configurations of antiferromagnetic. Therefore, to reduce the time, effort and computation Siriwardane et al., 2023 proposed a Cubic GAN model to generate ternary materials which are then screened using the ANN model as magnetic and non-magnetic as magnetic materials can be potential new spintronic materials after screening the magnetic material is further analyzed using DFT based stability. From 141 materials they were able to classify 45 materials as magnetic. In current work the part of building the classifier using ML and DL is validated with the stated accuracy and an effort is made to enhance the performance of the model. Several machine learning models are tested for this dataset to check model performance. Though in the reviewed paper, the ANN model was built using Keras and TensorFlow while in this work, the model is implemented using Pytorch, and hyperparameters were tunned using the Ray-Tunning technique. With lots of effort, I could achieve a 3% increment in accuracy and a slightly better performance report than stated in the reviewed paper. The model which is generated through experiment-II in this work has an input layer, hidden1, hidden2, an output layer, learning rate, weight decay and batch size for training the model is 112, 85, 29, 1, 0.0027, 0.001 and 729 respectively on split ratio as 80:10:10. Though the increment achieved is not that much significant still other ideas can always be thought off. In the current work, an effort to identify the names of features concerning data provided in atomic data is also tried to match by extracting data from the MP database using Pymatgen. But unable to match all the 56 features, out of 56 only 20 are properly matched in the current work. The name of the 42 attributes listed in Table 1, is from supporting Information provided in Siriwardane et al., 2023. The idea was to add some more attributes to the dataset after recognizing all the attributes so that model performance can be achieved still it remained incomplete due to insufficient information available.

1. **Future Work**

For discovering a new stable spintronic material machine learning and deep learning approach is outstanding. In this work, an effort to enhance the performance of the ML and DL-based models has been made. Though slight improvement (1%) is achieved in the performance of the ANN model built in this work, furthermore, ideas can be explored:

1. Effort can be further made to add more attributes related to the magnetism of elements or materials.
2. The transformers-based deep learning approach can be introduced to learn the sequential nature of materials so that new combinations of materials can be generated.

**References**

1. Siriwardane, Edirisuriya, Yong Zhao, and Jianjun Hu. "Data-Driven Deep Generative Design of Stable Spintronic Materials." (2023).
2. Puebla, Jorge, et al. "Spintronic devices for energy-efficient data storage and energy harvesting." *Communications Materials* 1.1 (2020): 24.
3. Vlaardingerbroek, Marinus T., and Jacques A. Boer. *Magnetic resonance imaging: theory and practice*. Springer Science & Business Media, 2013.
4. Manchon, Aurelien, et al. "Current-induced spin-orbit torques in ferromagnetic and antiferromagnetic systems." *Reviews of Modern Physics* 91.3 (2019): 035004.
5. Ramasamy, Karthik, Soubantika Palchoudhury, and Arunava Gupta. "Synthesis and Properties of Magnetic Chalcogenide Nanostructures." *Magnetic Nanomaterials: Fundamentals, Synthesis and Applications* (2017): 191-216
6. Zhao, Yong, et al. "High‐throughput discovery of novel cubic crystal materials using deep generative neural networks." *Advanced Science* 8.20 (2021): 2100566.
7. Jain, Anubhav, et al. "Commentary: The Materials Project: A materials genome approach to accelerating materials innovation." *APL materials* 1.1 (2013).