Identify Topological Material with Machine Learning

Junbo Zhu
Department of Physics
Massachusetts Institute of Technology
Cambridge, MA 02139
junbozhu@mit.edu

Abstract

Topological materials exhibit unique physics stemmed from the nontrivial topology in the electron bands, with potential applications for novel circuits. Topological database now contain tens of thousands of materials, screened by first principle and symmetry-based calculations. Here I attempt to train a classifier on database "Materiae" that predicts whether a material is topologically trivial or non-trivial given its crystalline info. Trained linear classifier and feedforward neural network gives 85% and 87% testing accuracy. Results of different optimizers and hyper parameter sets are discussed. Attempt via automatic machine learning function of H2O gives instead 91% test accuracy, and indicates that gradient boosting machine is better model. Github repo can be found at this link.

1 Introduction

Topological material, a specific class of single crystalline compounds, has become an important field in condensed matter physics due to the unique properties protected by the nontrivial topology of its electron bands[1]. Topological insulators (TI) are compounds which have insulating bulk, like a trivial insulator, but their edge or surface have protected conducting channels. Those dissipationless edge states may lead to a new generation of electronic circuits, especially a possible route for fault-tolerant quantum computation when interacting with superconductivity[2,3].

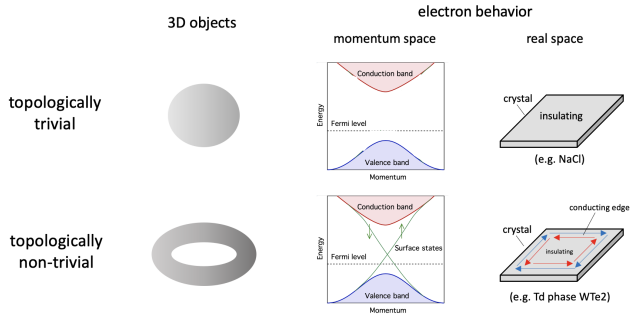

Figure 1: Illustration of topologically trivial and non-trivial classes. A sphere is trivial as a 3D geometric object, while a torus is non-trivial. In the case of insulators, trivial insulator is just gapped at Fermi level in its electron bands. But topologically non-trivial insulators has extra surface states going across the gap, which corresponds to conducting edge channels in real space.

Since the discovery of the first TI, searching for new topological materials has been a challenge. Identifying the topological nature (trivial or non-trivial) of a given compounds requires the first-principle computation of its electron bands and further the topological invariants. The recently

Preprint. Under review.

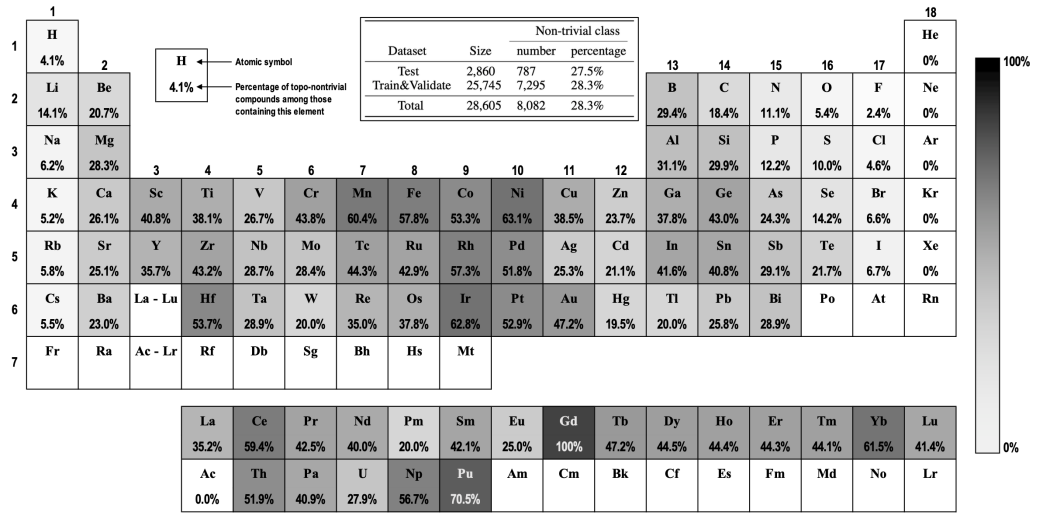

Figure 2: Atomic composition of topological non-trivial compounds in the dataset used in this report. The percentage of non-trivial compounds among those containing a given element is listed and also denoted by the grayscale of the cell. The darker it is, the larger the percentage. Blank cell refers that no compound in the dataset containing this element. Note that number under some elements may be misleading due to lack of data, e.g. only 3 compounds contain Gd and all of them are non-trivial.

established topological quantum chemistry[4,5], however, simplifies the procedure and accelerates the growth of topological material list. Up til now, people have computed $\sim 40,000$ materials and discovered $\sim 10,000$ non-trivial ones[6,7]. The scale of the database makes it possible to train a classifier via machine learning. In this report I attempt to train a machine learned binary classifier (trivial or non-trivial) given the general info of the crystalline material. Models including linear classifier, feedforward neural network and multiple models with automatic machine learning.

There are reported attempts on using machine learning to help identify the topological property. Some of them replace part of the calculation with machine learning, instead of replacing the whole calculation as proposed here, e.g. unsupervised learning of the deformations of electron Hamitonians[11], prediction of formation energy by machine learning instead of density function theory[12]. Others aim at different goals, e.g. use convolutional neural networks to analyze experiment raw data for topological property[13], represent certain topological electron states with neural networks[14].

One report that matches the same proposed problem as here is by Claussen, et al (2020)[9], who used gradient boosted trees to construct a machine learning model and achieved an accuracy of 90%. Moreover, they observed that "coarse-grained" chemical composition and crystal symmetry are the most important features in determining topology. This 90% accuracy is a practical goal for my neural network classifier.

2 Methods

2.1 Dataset

The dataset used here is from topological database "Materiae" [6]. There are 28,605 materials in total, 8,802 of which are topologically non-trivial. Class trivial is labeled as 0 while class non-trivial is labeled as 1. 2,860 materials (10%) out of the whole dataset is randomly split out as test dataset. The rest is used for training and cross-validation. The percentage of non-trivial class in the test dataset is 27.5%, close to percentage 28.3% in the whole dataset (see inset of Fig.2). Therefore, I regard the dataset is split without loss of generality.

To take a closer look at the dataset, I summarized the atomic composition of topological non-trivial compounds, displayed as the gray-scaled periodic table in Fig.2. The number under the atomic symbol is the percentage of non-trivial compounds within those containing this element. The darker

Table 1: Summary of the results.

Model	Hidden layer units	Optimizer	Optimal hyper parameter	Cut-off Epoch	Training		Validation		Testing	
					Averaged loss	Averaged accuracy	Averaged loss	Averaged accuracy	Loss	Accuracy
Linear classifier	\	SGD	lr=0.01, mo=0.9	16	0.378 (2)	0.827 (1)	0.38 (1)	0.827 (6)	\	\
	\	Adam	Default	16	0.375 (3)	0.829 (1)	0.38 (1)	0.828 (5)	0.349	0.850
Neural network	1 layer: 40 units	SGD	lr=0.01, mo=0.9	90	0.280 (2)	0.883 (2)	0.33 (2)	0.863 (8)	\	\
		Adam	Default	32	0.276 (3)	0.885 (2)	0.32 (1)	0.859 (6)	0.292	0.874
	2 layers: 80, 20	SGD	lr=0.01, mo=0.5	128	0.261 (3)	0.891 (2)	0.40 (6)	0.829 (3)	\	\
		Adam	Default	20	0.249 (7)	0.897 (3)	0.32 (2)	0.866 (6)	0.305	0.870
	3 layers: 80, 40, 20	SGD	lr=0.01, mo=0.9	50	0.24 (1)	0.901 (5)	0.32 (2)	0.86 (1)	\	\
		Adam	Default	18	0.230 (8)	0.906 (4)	0.32 (2)	0.870 (7)	0.316	0.872

AutoML (H2O)	Number of Models	Number of folds	seed	Training Loss	Training AUC	Testing Accuracy
	10	5	1	0.261	0.947	0.906
	30	10	1	0.243	0.954	0.913

Gray-scaled lines are optimized results for each model and method.

Cross-validation loss and accuracy listed are averaged value across all folds (number in brackets is standard deviation at last digit). The mini-batch size is 64 for both SGD and Adam. One epoch denotes a full iteration of the training dataset. Cut-off epoch is determined by when loss is saturated or overfitting problem arises. SGD parameter "lr" refers to learning rate and "mo" refers to momentum. Adam uses default hyper parameters, namely learning rate = $1e^{-3}$, betas = (0.9, 0.999), eps = $1e^{-8}$, weight decay = 0 and amsgrad = False.

AUC: area under the ROC curve; ROC: receiver operating characteristic curve.

the grayscale is, the larger the percentage. Some cells are left blank because no compound in the dataset containing the element. Note that number under some elements may be misleading due to lack of data, e.g. only 3 compounds contain Gd and all of them are non-trivial. Compounds containing transition metal and rare-earth have pronouncedly higher chance to be topological nontrivial. This matches the instinct that large spin orbit coupling from d and f electrons commonly play an important role in band inversion. Compounds consist lighter elements like NaCl are usually simple and trivial.

Symmetry of crystal lattice plays another important role in making a topological non-trivial material. For example, SnTe with mirror symmetry represents the class of topological crystalline insulator[1]. The complete symmetry operation of a given compound can be referred by its spacegroup number (230 in total that can exist in three dimensional world).

2.2 Feature Representation

The material info available from "Materiae" includes formula of the material, symmetry of the crystal structure indicated by spacegroup number; and number of atomic sites and electrons. I constructed feature representations (145 dimensions in total) as following:

1. Element composition:
   Weighted-one-hot representation of 109 dimensions corresponding to each element in the periodic table. Weight in jth dimension is equal to the fraction of atoms of jth element in the material. For example, the stereotype TI Bi_2Se_3 is represented as $\frac{2}{5}\hat{e}_{Bi} + \frac{3}{5}\hat{e}_{Se}$, while $BiSe_2$ is a trivial insulator and represented as $\frac{1}{3}\hat{e}_{Bi} + \frac{2}{3}\hat{e}_{Se}$
2. Element period and group:
   Weighted-one-hot representation of 7 periods and 20 groups, which refer to the rows and columns of the periodic table, i.e. number of electron shells and outer shell electron number. This spares our model from learning the structure of the periodic table. Weights are calculated the same way as element composition, e.g. Bi_2Se_3 is represented as $\frac{3}{5}\hat{e}_4 + \frac{2}{5}\hat{e}_6$ for period and $\frac{2}{5}\hat{e}_{15} + \frac{3}{5}\hat{e}_{16}$ for group. Weights are simply summed up if several elements in one material belong to the same period or group.
3. Crystal lattice symmetry:
   One-hot representation of the 7 categories (i.e. bravais lattice) of crystal spacegroup. There are 230 spacegroups in total, however, chemical formula plus its lattice category is usually enough to uniquely identify a given material. Therefore, only 7 feature dimensions is used to represent symmetry info.
4. Number of total atomic sites in the unit cell: standardized float number
5. Number of total electrons in the unit cell: standardized float number

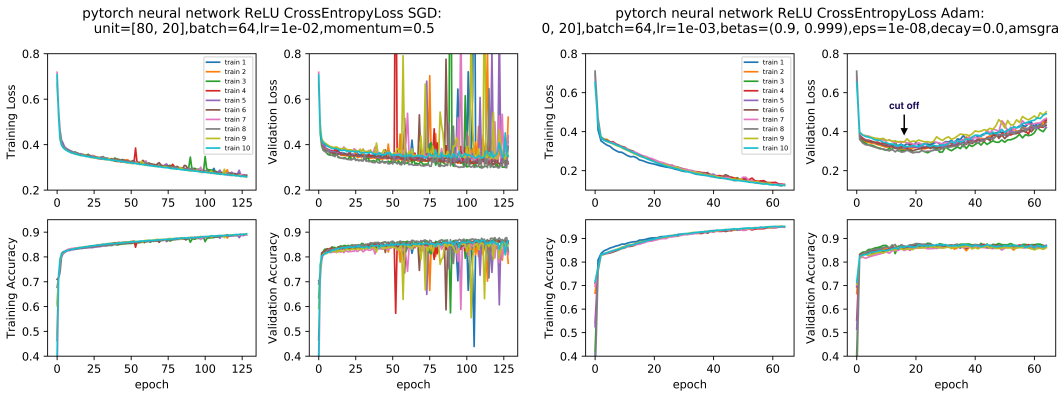

Figure 3: Comparison of training via SGD (left panel) and Adam(right panel). The neural network trained is the same that has 2 hidden layers with 80 and 20 units. Both SGD and Adam update with the size of mini-batch being 64. SGD uses learning rate of 0.01 and momentum of 0.5. Adam uses default parameters that are listed below Table.1. Adam clearly outperforms SGD.

2.3 Models and validation procedure

Pytorch is the main machine learning library used in this project. Github repo can be found at this link. Ten-fold cross validation is used to select optimal hyper parameters of a given neural network, as well as a reasonable cut-off epoch number. Afterwards, model is retrained with the complete train&validate dataset with the selected hyper parameters. The obtained model is then tested with the preserved test dataset. Cross entropy loss is used as loss function and the performance are evaluated by the prediction accuracy, i.e. the correct prediction rate.

As an attempt to compare more models on our problem, auto machine learning via H2O is also used on the same splitting of training and test dataset. The set ups attempted are summarized as below:

Models:

1. Linear classifier:
   Given the feature space being pretty sparse, linear classifier works. It's realized with a network without hidden layer, i.e. just a linear module with input and output dimension being 145 and 2.
2. Feedforward neural network:
   - Number of hidden layers: 1 to 3;
   - Number of units in hidden layers: typically 80, 40 or 20;
   - Activate function for hidden layers: ReLU;
   - Initialization of model weights follows a distribution centered at zero, with a standard deviation $= 1/\sqrt{n_i^l}$, n_i^l is the number of input units at layer l.
3. Auto machine learning (AutoML) by H2O toolbox:
   - Number of models trained and tuned: 10 or 30 models in total with 10-fold cross-validation.
   - Models scanned including: Gradient Boosting Machine (GBM), Extremely Randomized Trees (XRT), Distributed Random Forest (DRF), Deep Learning, as well as grid (hyperparameter) searching of GBM and Deep Learning models.
   - Stacked ensemble of all models trained is then tested on the test dataset.

Optimizers of linear classifier and feedforward neural network:

1. Mini-batch gradient descent with momentum: torch.optim.SGD(). Several learning rates and momentums are tested to find the better one.
2. Adam: torch.optim.Adam(). Default hyper parameters works well, therefore further fine tuning is not tried so far.

Table 2: AutoML Leader Board

Rank	model_id	logloss	AUC	Rank	model_id	logloss	AUC
	StackedEnsemble_AllModels_AutoML	0.243	0.954	15	GBM_grid__1_AutoML_model_8	0.299	0.931
1	GBM_grid__1_AutoML_model_4	0.243	0.954	16	GBM_grid__1_AutoML_model_6	0.312	0.926
	StackedEnsemble_BestOfFamily_AutoML	0.243	0.954	17	GBM_grid__1_AutoML_model_9	0.318	0.923
2	GBM_4_AutoML	0.259	0.948	18	GBM_grid__1_AutoML_model_3	0.325	0.918
3	GBM_grid__1_AutoML_model_2	0.258	0.948	19	DeepLearning_grid__2_AutoML_model_2	0.339	0.913
4	GBM_grid__1_AutoML_model_10	0.259	0.947	20	DeepLearning_grid__3_AutoML_model_2	0.336	0.910
5	GBM_grid__1_AutoML_model_7	0.263	0.946	21	DeepLearning_grid__3_AutoML_model_1	0.387	0.909
6	GBM_3_AutoML	0.268	0.945	22	DeepLearning_grid__1_AutoML_model_1	0.354	0.906
7	GBM_grid__1_AutoML_model_1	0.270	0.944	23	DeepLearning_grid__2_AutoML_model_1	0.406	0.906
8	GBM_2_AutoML	0.275	0.942	24	DeepLearning_1_AutoML	0.354	0.899
9	DRF_1_AutoML	0.287	0.940	25	DeepLearning_grid__2_AutoML_model_3	0.361	0.895
10	GBM_1_AutoML	0.281	0.940	26	DeepLearning_grid__3_AutoML_model_3	0.360	0.892
11	GBM_5_AutoML	0.280	0.940	27	DeepLearning_grid__1_AutoML_model_3	0.376	0.891
12	XRT_1_AutoML	0.298	0.936	28	DeepLearning_grid__1_AutoML_model_4	0.405	0.888
13	GBM_grid__1_AutoML_model_11	0.291	0.936	29	DeepLearning_grid__1_AutoML_model_2	0.393	0.886
14	GBM_grid__1_AutoML_model_5	0.297	0.932	30	GLM_1_AutoML	0.378	0.883

Max model number = 30, seed = 1, number of folds for validation = 10

AUC: area under the ROC curve; ROC: receiver operating characteristic curve

3 Results and Discussion

A summary of the results is shown in Table.1. Gray-scaled lines are optimized results for eachmethod. As described above, optimal hyper parameters and a cut-off epoch are determined during cross-validation. Then they are used to retrain the model with the whole train&validate dataset. Finally the retrained model is tested with the preserved test dataset.

3.1 Linear Classifier

The trained linear classifier reaches surprisingly high accuracy up to 83% both for training and validation and gives 85% testing accuracy. In comparison, the best neural network gives 87% and ref[9] reported 90% with their gradient boosted trees model. The high performance of linear classifier is understandable because the feature space are very sparse due to those dominant one-hot representations. One can take a look at the weights of trained linear classifier. The top 3 features with the highest weights in favor of the non-trivial class are: No.1. containing elements from group VI (group 16); No. 2. containing Mo; No.3. containing Bi. On one hand, indeed Bi, due to its large spin orbit coupling, is one common element seen in topological material. Moreover, the first discovered TI family Bi_2Se_3 and Bi_2Te_3 consist of Bi and one element from group VI. On the other hand, if one goes back to Fig. 1 he will notice that only 29% of the compounds containing Bi are non-trivial (the value is 28% for Mo case), far lower than other well-known "topological favoring" elements like Ir (63%) and Sn (41%). Therefore, the large slope of trained classifier on Bi and Mo doesn't directly reflect the element-wise view of our dataset. Instead, combining element-wise information and lattice symmetry info, even a linear classifier can do a good job in topological material prediction, i.e. 85% accuracy compare to 28% of a random guess.

3.2 SGD vs Adam

Due to personal learning purpose, I performed both SGD and Adam as optimizers. Adam turns out to always out perform SGD in training our problem, both more efficient and more effective. Figure.2 gives a comparison of SGD (left panel) and Adam (right panel) training the same neural network that has 2 hidden layers with 80 and 20 units respectively. Adam reaches saturation of validation accuracy soon after 20 epochs and then starts to overfits which can be tell by the upturn of validation loss curve. SGD, however, need much more epochs to reach saturation and meanwhile cannot get rid of "spikes" in both loss and accuracy. Reducing parameter momentum of SGD can improve the smoothness of training loss and accuracy but has no help on validation curve. I believe this is due to the specific landscape of loss function in feature space, that momentum only is not enough to help settle down around minimum. Adam combines the ideas of momentum and adaptive step size and therefore has much better performance

3.3 Auto Machine Learning

The leader board of AutoML trained models are listed in Table.2. The stacked ensemble of all models trained gives a training AUC (area under the ROC curve, ROC be receiver operating characteristic curve) of 95% and a testing accuracy of 91%, also listed in Table.1. The leader models obtained are grid searching of gradient boosting machine (GBM), see Table.2. Therefore it's consistent with the result of Claussen, et al (2020)[9] who used gradient boosting trees and obtained 90% accuracy, although on a different database from me.

The stacked ensemble of all models gives several percent higher accuracy than the feedforward neural network I trained. Note that, however, the performance of Deep Learning model trained by H2O AutoML with grid searching is similar to my trained neural network. Therefore, I regard the hyper parameter tuning during my training is successful and further improvement of prediction accuracy is limited by neural network model itself. The specific reason why GBM outperforms neural networks remains further inspection.

4 Conclusion

In this report, I attempt to train a binary classifier (trivial or non-trivial) given the general info of the crystalline material. Dataset is grabbed from topological database "Materiae"[6], containing 28,605 entries with 28.3% belonging to non-trivial class. Ten-fold cross validation is used for training with 10% of dataset reserved for testing. Trained linear classifier and feedforward neural network gives 85% and 87% testing accuracy, while stacked ensemble of all models trained by AutoML reaches 91%. The leader model is searching grid of GBMs. This result is consistent with the recent report[9] on a different database[7].

For future interest, one may apply our trained model to materials that not yet included in the topological database, or even new compounds that has not been realized in lab. This will be significantly time saving compare to traditional ways.

References

[1] M. Z. Hasan and C. L. Kane, Rev. Mod. Phys. 82, 3045 (2010)

[2] V. Lahtinen, J. K. Pachos, SciPost Phys. 3, 021 (2017)

[3] Z. Wang. Topological quantum computation (No. 112). American Mathematical Soc. (2010)

[4] B. Bradlyn, L. Elcoro, J. Cano, M. Vergniory, Z.Wang, C. Felser, M. Aroyo, and B. A. Bernevig, Nature 547, 298 (2017).

[5] T. Zhang, Y. Jiang, Z. Song, H. Huang, Y. He, Z. Fang, H. Weng, and C. Fang, Nature 566, 475 (2019).

[6] Topological material data base #1: `http://materiae.iphy.ac.cn/`

[7] Topological material data base #2: `https://topologicalquantumchemistry.org/#/`

[8] Ching-Kai Chiu, Jeffrey C.?Y. Teo, Andreas P. Schnyder, and Shinsei Ryu Rev. Mod. Phys. 88, 035005

[9] N. Claussen, B. A. Bernevig, and N. Regnault, Phys. Rev. B 101, 245117 (2020)

[10] General crystal data base: `https://icsd.fiz-karlsruhe.de/index.xhtml`

[11] M S. Scheurer and R. Slager, Phys. Rev. Lett. 124, 226401 (2020)

[12] Y. Jiang, et al. npj Comput Mater 7, 28 (2021)

[13] W. Lian etc, Phys. Rev. Lett. 122, 210503 (2019)

[14] D. Deng, X. Li, and S. D. Sarma Phys. Rev. B 96, 195145 (2017)