Topological Data Analysis Based Feature Selection for Predicting Fatigue Strength of Steel Using Machine Learning

M A Joshi and D L Joshi
1Department of Applied Science and Humanities, MPSTME NMIMS, Shirpur, India
2Department of Computer Engineering, MPSTME NMIMS, Shirpur, India
mlnjsh@gmail.com

Abstract. Routine Machine learning methodologies are based on Statistics and traditional data analysis methods, which may account for loss of some important information from the data during analytical process. So we need a new set of tools and algorithms that can effectively make most of such information. This paper investigates the implementation of new and growing data analytics technology known as topological data analysis (TDA). We use Mapper algorithm from TDA on data taken from public domain database of National Institute for Material Science (NIMS), for selecting key process parameters or features that impacts fatigue strength of steel. Mapper algorithm outputs a network graph called topological network that captures the important connectivity information and complex interactions between the clusters present in the data where traditional methods fail to capture such interactions. We analyse the shape of topological network obtained from mapper for selecting important features that impact the fatigue strength of steel. We then use XG-Boost prediction model with tuned hyper parameters using various tuning algorithms, to evaluate the impact of the selected features. This approach of data analysis and modelling helps to monitor and control the process in a more cost-effective manner. The experimental results have successfully proved the efficacy of TDA and machine learning (ML) and hence developed the predictive model for the same.

Keywords: TDA, Mapper, Feature Selection, Material Informatics, Machine Learning, XG-Boost, Regression, Steel, Fatigue, Processing-Property Linkages

1. Introduction
Fatigue strength plays major role for design and failure analysis of mechanical components, and serves as a basic data required for the same. Reports shows that fatigue is responsible for over 90% of all mechanical failures of structural components [1]. Enfeebling outcomes of fatigue failures with high cost and time for fatigue testing, automating prediction of fatigue life is of utmost importance not only for field of materials science but also for mechanical engineering i.e. it is important to have a prediction model that predicts fatigue strength of steels. In this work we use large number of heat treatment process and composition parameters for prediction. Most of the conventional methods failed to capture the complex interaction between the various input variables, so we give an attempt to use TDA which is proven to be powerful way to capture connectivity information between the input parameters [2, 3]. This study serves as an introductory guide for potential researchers in this field. The scope of this paper includes application of ML and TDA methods together to predict fatigue strength of steels using composition and processing parameters. In TDA we use mapper algorithm that gives network graph which is a powerful and compressive representation of a high dimensional complex dataset [4, 5].
Plotting the network provides a direct and insightful visualization of the underlying ‘data shape’, which can help to explore hidden patterns of the data. Nodes in the network correspond to groups of similar samples that have been clustered together. An edge between two nodes means that there are shared/common samples between the two nodes. Both large scale global structure and local group of nodes can be revealed in the network. Furthermore, mapping a specified target variable onto the network, using a color scheme, can show how the target variable change along with the network. Therefore, plotting a TDA network and coloring the network with interested target variables is the first step to understand the data. Once a TDA network has been constructed, network-based enrichment analysis can be performed for target variables, or properties of the studied data samples. We use the spatial analysis of functional enrichment (SAFE) for network enrichment analysis. This algorithm is originally developed as a systematic method for annotating biological networks and examining their functional associations. [6]. Once SAFE scores are calculated for target variable, we selected and rank the features that contribute for prediction of fatigue strength, on newly selected variable we apply XG-Boost which is modification of Gradient Boosting algorithm. The details of Gradient Boosting algorithm can be found in [7]. We use XG-Boost with tuned hyper parameters using various algorithms like Greed Search, Random Search, Genetic algorithm, differential Evolution. The details of XG-Boost can be found in [8]. We use RMSE, MAE and R$^2$ to evaluate the performance of our model on test data.

2. Data And Methods

2.1 Data
Data is the most important part any research, correct and accurate data endeavour fruitful results. In this research, Fatigue dataset for steel is used for predicting fatigue strength. It is accessed from National Institute of Material Science (NIMS) MatNavi [9]. The following table shows the data in detail.

Table 1: Dataset Description

| Contents of data          | Input parameters                                                                 |
|--------------------------|----------------------------------------------------------------------------------|
| Chemical composition     | - %C, %Si, %Mn, %P, %S, %Ni, %Cr, %Cu, %Mo (all in wt. %)                       |
| Upstream processing details | - ingot size, reduction ratio, non-metallic inclusions                           |
| Heat treatment conditions | - temperature, time and other process conditions for normalizing, through-hardening, carburizing-quenching and tempering processes |
| Mechanical properties    | - YS, UTS, %EL, %RA, hardness, Charpy impact value (J/cm2), fatigue strength      |

The 437 data instances include 371 carbon and low alloy steels, 48 carburizing steels, and 18 spring steels. This data pertains to various heats of each grade of steel and different processing conditions.

There are 25 input features and 1 target feature (Fatigue Strength) which pertain to rotating bending fatigue tests at room temperature conditions. Total observations are 437. In this study fatigue strength of steel is to be predicted using given input parameters.

2.2 Methods
Figure 1 below illustrate the overall approach. The first step is to pre-process the raw data for consistency using domain knowledge [1]. We start with data and pre-processed data obtained from [9, 10]. The pre-processed data is then input to Mapper algorithm which outputs a network called topological network. Based on shape of network and colouring of network with respect to target variable Fatigue strength SAFE scores are calculated for each node, based on SAFE score the features are selected as they are ranked. The new set of features are then selected for final study, we apply XG-Boost algorithm for prediction, finally effectiveness of model is tested on test set using performance measures like RMSE, MAE and R$^2$. Details of each of the 8 stages is represented below.
Figure 1. Pipe line for analysis and modelling using TDA and Machine Learning

3. TDA and MAPPER
TDA is branch of applied mathematics. TDA uses techniques from algebraic topology. Topology is very old branch of mathematics which studies shapes. TDA revolves around the principle that “Data has shape and shape has meaning”. Natural data offers hard challenges. Extracting information from noisy, incomplete and high-dimensional datasets is generally challenging. To analyse such data, TDA provides magical framework in a manner that it does not depend of choice of particular metric and it also provides dimensionality reduction and it is immune to noise.

In Mapper, we construct a network graph or in general (Simplicial complex which is generalized version of graph) from data in a way that captures topological features of the data. Mapper is visual representation of data in the form of cluster like that estimates various complex connections between the input parameters and those network clusters can colour according to various target variables, in our case target variable is Fatigue strength of steel. Such visual exploration of the data that can often reveal new insights of the data, it is an unsupervised method. Other methods cannot reveal such new insights.

Informally, the Mapper algorithm works by performing a local clustering based on given clustering method guided by a projection/filter function based on given filter function. The steps are as follows:

1. When we get a real world dataset, first we need to choose a distance metric to describe the distance between each pair of points. For example, a three-dimensional Euclidean distance metric. It is not necessary to use the Euclidean metric Predefined metric can also be used. The defined metric will be used by a filter function for data projection (or dimension reduction).

2. Next, Mapper projects the original data cloud to a low dimensional space, by using a filter function, which can be PCA, MDS, t-SNE, or any other dimension reduction method. In our case, the filter function is PCA with two components, in which the data points are coloured by the filter function. Different filters can also be combined to form lens, or a low dimensional space. Usually, Mapper uses a two-dimensional space of projected data.
3. After filtering, the projected dataset is binned into groups (hypercube) of data points, by a cover, which is a collection of intervals along the filters. A cover can be specified via two resolution parameters (resolution: number of intervals, and overlap: percentage of overlap between adjacent intervals).

4. The final step of Mapper is clustering and network construction. Clustering is performed on each bin/hypercube of data points to identify clusters of data points in the original data space, rather than in the projected data space. Network construction is based on the clustering results: each cluster forms a node in the network, and edges are made between two nodes if they have common samples shared by their corresponding clusters. Therefore, the resulted TDA network, as shown in figure 2, is a compressive network representation of the underlying ‘data shape’, which is an abstract shape of a circle in this case.

Figure 2: Mapper Implement of circle (a) Circle (b) Each data point on circle projected on y-axis (Lenses/Filter) (c) Filter is divided into Bins or number of overlapping intervals and then coloured according to some colour function (d) Taking Inverse –Image of coloured bins forming clusters each cluster is treated as node of graph connected with edges capturing connectivity information showing there by compressed representation of circle.

Our Approach A.

1. Bin the Fatigue strength into three bins (224.035, 546.667], (546.667,868.333],(868.333,1190.0]
2. Label the bins 0, 1, 2 respectively.
3. Split the whole data into input features and target features.
4. Apply mapper algorithm to our dataset with target variable as fatigue strength labelled as 0,1,2.
5. Construct mapper graph by trying various parameters of interest we have selected the best as per in table 2.
6. Try colouring nodes of graphs using our target variable Fatigue strength.
7. Graphs are also coloured using single proportion levels.

The details of the approach is shown in the following diagrams. We use Kepler-mapper python implementation/Scikit-tda library. The details can be found in [13, 14].
| Parameter          | Value                                                      |
|--------------------|------------------------------------------------------------|
| Projection 1       | Isolation forest (That detects the anomaly in the input data) |
| Projection 2       | L2 norm                                                    |
| Projection         | [Projection1,Projection2]                                  |
| Clustering algorithm | K-means (with 5 clusters)                                 |
| Colour function    | Proportion of y-class labels                               |
| Percent overlap    | 70                                                         |
| Number of bins     | 15                                                         |

Table 2: Mapper Parameters

Topological Network Representing the Steel Fatigue Dataset:

Figure 3: Topological Network Representing dataset Fatigue strength of steel with Graph Node distribution and information of network.

Following Diagram shows the same network with different clusters encircled, we also see interesting flares shown in blue circles, which is major finding of this work. Deep analysis of these flares and study
of deep interaction between the circled parts of this network is beyond the scope of this paper and can be incorporated in the future research.

Figure 4: Topological Network Representing dataset Fatigue strength of steel with clusters & Flares

Figure 5: Topological Network Representing dataset Fatigue strength of steel with target variable distribution
Figure 6: Nodes are coloured according to the production of fatigue strength
4. SAFE algorithm and Feature Selection

Once a TDA network has been constructed, network-based enrichment analysis can be performed for target variables, or properties of the studied data samples. We use SAFE for network enrichment analysis. This algorithm is originally developed as a systematic method for annotating biological networks and examining their functional associations.

The SAFE algorithm takes the following steps to calculate an enrichment score (SAFE score) of each node in a network for a given target variable:

1. Input TDA network graph and Target variable
2. For each node \( u \) in the network graph, SAFE defines a local neighbourhood of \( u \) by identifying any other nodes that are closer than or equal to a maximum distance threshold \( d \) to \( u \). Node distance is measured via shortest path length between nodes. By default, the maximum distance threshold \( d \) is set to be equal to the 0.5th percentile of all pairwise node distances in the network.
3. For each neighbourhood, SAFE sums the values of neighbour nodes for a target variable as an observed neighbourhood score \( S \). The score is then compared with the distribution of permuted neighbourhood scores obtained by randomly shuffling the target variable among nodes in the network. The significance of enrichment \( P \) is determined as the probability that a random observation from the distribution will fall into the interval between the observed neighbourhood score \( S \) and the largest value of the permuted scores. The neighbourhood significance of enrichment \( p \) is converted into a neighbourhood enrichment score \( O \), named SAFE score, as below (normalized to a range from 0 to 1):

\[
P_n = \frac{m}{I}
\]

Where \( m \) is the number of times a observed value is greater than or equal to shuffled values; \( I \) is the number of shuffles;

\[
O_n = \frac{-\log_{10}(\max(P_n, \frac{1}{I+1}))}{-\log_{10}\frac{1}{I+1}}
\]

Where \( I \) is the number of shuffles; \( P \) is the neighbourhood significance of enrichment of node \( n \); and \( O \) is the neighbourhood enrichment score (SAFE score) of node \( n \). Random shuffle is performed independently for each target variable when there are more than one.

A node is considered to be significantly enriched under a \( p \)-value threshold of 0.05, which can be translated to a threshold of SAFE score by:

\[
O_n \geq \frac{-\log_{10} 0.05}{-\log_{10}\frac{1}{I+1}}
\]

Filter and rank target variables using number of significant nodes or sum of SAFE score of significant nodes. We use tmap package implemented in python to calculate safe score, the details can be found in [15]
Our Approach B.

1. Once the mapper graph is constructed we use SAFE score to select important variables
2. According to SAFE scores We see that in the following diagram S, THQCr, Ni, dA, P, RedRatio, DT, SL.No, CU, Tt, Cr, NT, CT, Mo, Mn, TT, DT DC, DB are important features

Figure 7: feature selection and ranking using SAFE Scores based on topological network with respect to target variable as steel fatigue strength
5. Gradient Boosting (GB) Algorithm

GB is a powerful machine learning algorithm that can do regression, classification, and ranking. Gradient Boosting has two steps: Gradient Descent and Boosting. GB produces a prediction model by combining a collection of weak prediction models, also known as an ensemble of weak learners, basically boosting Models comprises ensemble of base learners with Low Variance and High Bias. The base learners are basically shallow decision trees. The core idea of Boosting is, given training data \( D_{train} = \{ x_i, y_i \}_{i=1}^n \) we train our first model say \( M_0 \) on whole of our training data. \( M_0 \) needs to have high bias and low variance. \( M_0 \) could be shallow decision tree or any other model. High bias means high training error. Let \( M_0: y = h_0(x) \). Now we define an error at \( i^{th} \) data point to be simply \( Error_i = y_i - h_0(x_i) \). Now we have \( \{ x_i, y_i, error_i \}_{i=1}^n \) for each data point \( i \) for which \( \leq i \leq n \). Now we will construct another model \( M_1: y = h_1(x) \) with new training data \( \{ x_i, error_i \}_{i=1}^n \), now our final model will become \( F_1(x) = \alpha_0 h_0(x) + \alpha_1 h_1(x) \). We may stop or we can go ahead as \( F_2(x) = F_1(x) + \nu \alpha_2 h_2(x) \) etc. Here we are building models in a stage wise fashion with optimizing the arbitrary differentiable loss function. Implementing gradient boosting in Python or in R takes lot of time and space, so we need scalable/parallelizable, quick to execute tool to work. eXtreme Gradient Boosting or XGBoost is a library of gradient boosting algorithms optimized for modern data science problems and tools. XGBoost does Parallel Computing, Regularization, Enabled Cross Validation, Missing Values, Flexibility, Availability, these problems and typically outperforms other algorithms. More about XGBoost can be found in [8]. We give the pseudo code for gradient boosting as follows

1. Initialize \( f_0(x) = \arg \min_\gamma \sum_{i=1}^N L(y_i, \gamma) \)

2. For \( m = 1 \) to \( M \):
   a. For \( i = 1, 2, \ldots, N \) compute the residuals

   \[
   r_{im} = - \left[ \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right] \bigg|_{f=f_{m-1}}
   \]

   b. Fit a regression tree to the target \( r_{im} \) giving terminal regions \( R_{jm}, j = 1, 2, \ldots, J_m \).

   c. For \( j = 1, 2, \ldots, J_m \) compute the prediction in the region \( R_{jm} \)

   \[
   \gamma_{jm} = \arg \min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma).
   \]

   d. Update \( f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm}) \), where \( I(x \in R_{jm}) = 1 \) if \( x \in R_{jm} \), otherwise \( I(x \in R_{jm}) = 0 \).

3. Output \( \hat{f} = f_M(x) \).

Our Approach C.

1. Once important features are selected using SAFE scores, we use modified version of Gradient Boosting called XG-Boost
2. Hyper-parameters of XG-Boost are listed in the table 3
3. We optimize set of hyper-parameters of XG-boost using following optimization algorithms.
4. Grid search, Random search, genetic algorithm, differential evolution.
5. We describe details of Grid search in this paper in table 4
6. Grid search does the best job, it can be seen in table 5 and 6 reporting RMSE 23.6631, MAE 15.1262 and R^2 0.9888
7. Finally, we have plotted the predicted vs. actual
### Table 3: Hyper-Parameters in XG-Boost

| Parameter      | Description                                      |
|----------------|--------------------------------------------------|
| Nrounds (M)    | Total number of trees                            |
| Eta (0 < η ≤ 1)| Learning rate                                    |
| Max_depth      | Maximum depth of tree                            |
| Min_child_weight| Minimum number of samples in each node. The larger the value, the more conservative algorithm will be |
| Subsample (0 < ω ≤ 1) | Proportion of samples used in each round, ω=1 indicates using all samples in the training set |
| Colsample_bytree (0 < v ≤ 1) | Proportion of variables used in each round, v=1 indicates using all variables |
| Lambda (λ ≥ 0) | Ridge regularization (L2) term on weight         |
| Alpha (α ≥ 0)  | Lasso regularization (L1) term on weight          |

### Table 4: Grid search of best hyper-parameters for XG-Boost

| Parameters                                      | Description                      |
|------------------------------------------------|----------------------------------|
| Number of trees (around)                       | [min = 50, max = 500, step=50]   |
| Learning rate (eta)                            | [min = 0.01, max = 1, step=0.5]  |
| Lambda # L2 Regularization (Ridge Regression)  | [min = 0.01, max = 1, step=0.5]  |
| Alpha # L1 Regularization (Lasso Regression)   | [min = 0.01, max = 1, step=0.5]  |

### 5.1 Prediction and Results Using XG-Boost

#### Table 5: XG-Boost performance with hyper-parameter tuning algorithms on train set

| Method                      | RMSE   | MAE    | Rsquared | nrounds | eta   | lambda | alpha | Processing Time |
|-----------------------------|--------|--------|----------|---------|-------|--------|-------|-----------------|
| Grid Search                 | 29.76842 | 19.69660 | 0.97498  | 50      | 1.00000 | 0.10000 | 1.00000 | 3 mins          |
| Random Search               | 29.75865 | 19.53965 | 0.97491  | 48      | 0.32730 | 0.09358 | 0.04431 | 1 mins          |
| Genetic Algorithm           | 30.30175 | 19.95909 | 0.97416  | 887     | 0.00054 | 0.09803 | 0.08894 | 12 mins         |
| Differential Evolution      | 31.23318 | 20.76231 | 0.97271  | 71      | 9.07471 | 0.04417 | 0.03711 | 31 mins         |

Note: Summary statistics obtained using 5-fold cross validation repeated 3 times. Grid and Random Search performed with adaptive resampling.

#### Table 6: XG-Boost performance with hyper-parameter tuning algorithms on test set

| Method                      | RMSE   | MAE    | R-squared |
|-----------------------------|--------|--------|-----------|
| Grid Search                 | 23.6631 | 15.1262 | 0.9888    |
| Random Search               | 24.3655 | 15.4806 | 0.9881    |
| Genetic Algorithm           | 24.7095 | 15.8165 | 0.9879    |
| Differential Evolution      | 23.8552 | 15.5625 | 0.9886    |
Figure 8. Plotting predicted vs. actual for all optimization algorithms
Conclusion: In this paper we have successfully applied topological data analysis method called mapper to select key process variables that affects the fatigue strength of steel using SAFE scored method which is used for enrichment analysis in biological network. Further we have built prediction model XG-Boost as regression model based on selected parameters, which shows pretty good accuracy. So we have effectively utilize TDA and Machine learning to build prediction model. Further the network graph can be analysed into more details, for example in this paper we did not talk about the clusters and the flares, which can be explored in a great detail to get more insight of data as an extension of this research in future.

References
[1] Dieter GE: Mechanical Metallurgy. Mc Graw-Hill Book Co. 3rd edition, ISBN: 0–07-016893–8 26. Deshpande PD, Gautham BP, Cecen A, Kalidindi S, Agrawal A, Choudhary (2013) Application of Statistical and Machine Learning Techniques for Correlating Properties to Composition and Manufacturing Processes of Steels. In 2nd World Congress on Integrated Computational Materials Engineering. John Wiley & Sons, Inc.; 1986:155–160. ISBN: 9781118767061 ISBN: 9781118767061
[2] G. Carlsson, "Topology and data", Bull. Amer. Math. Soc., vol. 46, no. 2, pp. 255-308, 2009
[3] Edelsbrunner H, Harer J. Computational topology: an introduction [M]. American mathematical Soc., 2010
[4] Lum P.Y, Singh G, Lehman A., et al. Extracting insights from the shape of complex data using topology[j] Scientific reports , 2013, 3:serp01236
[5] Gurjeet Singh, Facundo Mémoli and Gunnar Carlsson, Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition, Eurographics Symposium on Point Based Graphics, European Association for Computer Graphics, 2007, pp. 91–100
[6] Baryshnikova A. Systematic functional annotation and visualization of biological networks [J]. Cell systems, 2016, 2(6): 412–421.
[7] Friedman, J. H. Greedy function approximation: a gradient boosting machine. Ann. Stat. 29, 1189–1232 (2001).
[8] Chen, Tianqi and Guestrin, Carlos A Scalable Tree Boosting System Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
[9] URL: National Institute of Materials Science. http://smds.nims.go.jp/fatigue/index_en.html.
[10] Agrawal, A., Deshpande, P.D., Cecen, A. et al. Exploration of data science techniques to predict fatigue strength of steel from composition and processing parameters Integrating Materials (2014) 3: 90.
[11] W. Guo and A. G. Banerjee, “Toward automated prediction of manufacturing productivity based on feature selection using topological data analysis,” 2016 IEEE International Symposium on Assembly and Manufacturing (ISAM), Fort Worth, TX, 2016, pp. 31-36.
[12] Rajan K, Suh C, Mendez P: Principal component analysis and dimensional analysis as materials informatics tools to reduce dimensionality in materials science and engineering. Stat Anal Data Min 2009, 1: 361–371. 10.1002/sam.10031
[13] van Veen et al., (2019). Kepler Mapper: A flexible Python implementation of the Mapper algorithm. Journal of Open Source Software, 4(42), 1315
[14] Hendrik Jacob van Veen, Nathaniel Saul, Eargle, David, & Sam W. Mangham. (2019, October 14). Kepler Mapper: A flexible Python implementation of the Mapper algorithm (Version 1.3.3). Zenodo. http://doi.org/10.5281/zenodo.3485851
[15] Tianhua Liao, Yuchen Wei, Mingjing Luo, Guoping Zhao, Haokui Zhou tmap: topological analysis of population-scale microbiome data bioRxiv 396960;