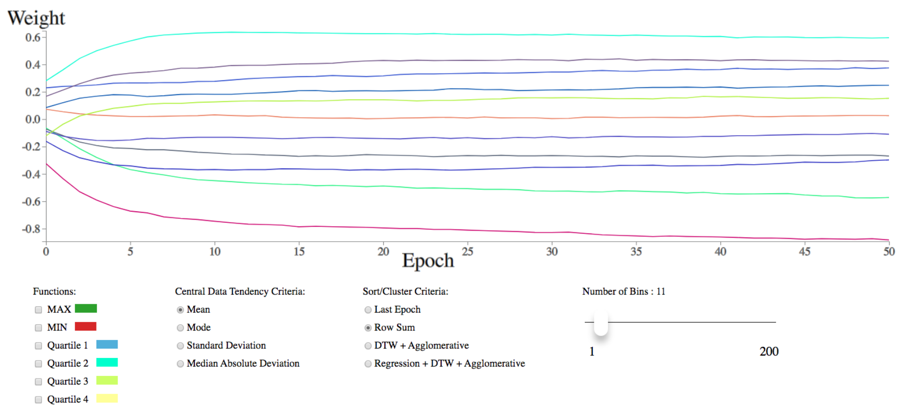
Visually Explaining the Weight Distribution of Neural Networks Over Time

Sahil Pasricha
Universität Konstanz
4 PUBLICATIONS **0** CITATIONS

Visually Explaining the Weight Distribution of Neural Networks Over Time

Sahil Pasricha



Functions:
- ☐ MAX 🟩
- ☐ MIN 🟥
- ☐ Quartile 1 🟦
- ☐ Quartile 2 🟦
- ☐ Quartile 3 🟩
- ☐ Quartile 4 🟨

Central Data Tendency Criteria:
- ● Mean
- ○ Mode
- ○ Standard Deviation
- ○ Median Absolute Deviation

Sort/Cluster Criteria:
- ○ Last Epoch
- ● Row Sum
- ○ DTW + Agglomerative
- ○ Regression + DTW + Agglomerative

Number of Bins : 11

1 200

Fig. 1. A summarized view of features offered by the proposed visualization method. The pattern indicates a higher degree of learning in initial training steps as compared to later steps.

Abstract—We present an interactive visualization method of the statistical distribution of weight over learning steps. A representation that assists the user with insights about the ML process, expediting model optimization procedure with measured calculations to improve the accuracy of a model. This visualization assists the user to understand the quality of the model and leverage the insights to further optimize the ML algorithm. We investigate the design space of explanation processes based on core factors of machine learning algorithms to present the learning process using the distribution of weights over the time steps. To support this, we analyse visualization parameters and summary statistics to present the data in a condensed form without compromising the potentially critical information in context. Specifically, we examine complex time series, clustering and sorting methods. Based on the learning derived from our analysis, we introduce distinctive combinations of Central Data Tendency with ML algorithms to present different perspectives for data exploration. Furthermore, to demonstrate the utility of the method, we evaluate our method with different models and data sets to observe interesting patterns. The most interesting insights that our method presented, comes from a model with bad accuracy. By observing the visualization, we analyse that the model finds perfect weights which represent train data and this happens in very initial training steps. In conclusion, this model requires more training data to optimize accuracy.

Index Terms— Visualization, Interactive Machine Learning (IML), Deep Neural Networks (DNN), Explainable AI, Visual Analytics, Interpretability, Explainability, Central Data Tendency (CDT), Dynamic Time Warping, Time Series Analysis

1 Motivation

With technology touching our life closer than ever before we have become heavily dependent on machine made decisions. We use it for varied tasks ranging from recognizing our face over unlocking smartphone to landing a 70 ton aircraft. Along with the growth in the frequency of AI being used in everyday life, the role of these machine made decisions have witnessed a surge in recent times. Given that, popular AI systems have already achieved considerable accuracy, what makes it susceptible is its inability to explain decisions it didn't make right.

Presently in a typical scenario, users apply a random algorithm for machine learning and iterate the process until it reaches desirable accuracy [6]. The fact is, this black box methodology works quite often but not always. But the majority of applications of AI are not mission critical and those which are, have already started working on Explainable Artificial Intelligence (XAI) to further dig into the model to understand its core. This application mainly includes autonomous driving, satellite imaging and autopilot for drones [15].

Speaking of XAI, modern deep learning platforms including TensorFlow [5] provide libraries to generate a low-level dataflow chart that supports a diversity of learning algorithms. While these graphs can be suitable for models that are typically small but as the size grows, these dataflow graphs present thousands of complex, low-level operations that may overwhelm the user with too much information.

Apart from these conventional tools, there has been considerable progress in the direction of customized XAI methods that take the explanations to various degrees and directions ranging from explaining the feature importance by visual maps (saliency maps) [28] for explaining individual prediction (LIME) [24]. The recent research over interactive and Explainable Machine Learning [12] [10] provides an entry point to classifications of existing XAI methods. Also, a survey paper [30] sheds light on the interpretability of these models. Something common in almost all of the existing XAI models is the dependency and integration. All of them are designed to work with a definite

Sahil Pasricha, Thilo Spinner, Hanna Schäfer, and Daniel A. Keim are with the University of Konstanz.
E-mail: {sahil.pasricha, thilo.spinner, hanna.schaefer, keim}@uni-konstanz.de.

domain, data or model and do not offer plug and play implementation with active ML developing and debugging process. None of them seem to be designed for an ML expert who typically works on ML models from varied domains, has different structures, and trains on diverse data sets.

This motivates us to research in the direction of a domain, data and model independent XAI method that facilitates an ML expert with ML algorithm's quality analysis through a visual exploration of its learning process, whose patterns may further reveal additional interesting insights. In addition, we not only want to describe the theoretical workflow but also deliver system implementation. Because the foundation of our proposed model are core feature of almost every ML algorithm, it is comparatively quick to implement. Our visualization method provides an interactive analysis of the learning process of an ML model.

This design study[26] began with evaluating fundamental features to determine the critical insights they could offer about the model. The next step was to research methods to present that in summarized and meaningful ways without information loss, this included evaluating basic methods like mean, median for central data tendency and introduced novel combination of sorting, clustering and time series algorithms to form bins/clusters that have more homogeneous data. To support exploration, we employed interactive zoom, log scale, adaptive binning and flexibility over bin count. Finally, we evaluated the usability of our method on specific data sets and analyzed how the revealed patterns can support a user in establishing a hypothesis about the quality of the model and explore possible ways to enhance the accuracy of the model through rich interactions.

2 Related Work

To devise a novel XAI method, we analyse existing research on XAI models, Information Visualization and survey papers that compared XAI methods on multiple scales. We also studied research analysing the impact of perturbation weights of an NN model [8].

We start with understanding the features and scope of existing XAI models[23]. Information flow [32] is one such method offering an overview of dataflow between groups of operations exhibiting details of its nested structure on demand. In contrast, LIME [24] focus solely upon explaining features and ANCHORS [14] uses novel model-agnostic explanations based on if-then rules, called anchors. By visualizing model performance while enabling direct data inspection, model tracker [8] offer direct error examination and debugging empowering user with useful controls right from GUI. Another technique called Layer-wise Relevance Propagation [9] explains the model's outcomes by decomposition. It reverse-engineers the output using local redistribution rules until it assigns a relevance score. Survey papers [12, 30] were critical to understand the dependency of these models. All of these models have used complex methods to derive important insights from the model by pondering upon the usability of any aspect of the model (weights, layers, structure) by harnessing its relation to other aspects and impact on output. This also inspire us to experiment methods (discussed in later section) that are novel to XAI methods.

To further broaden our knowledge on usage of visual analytics to understand the concepts of [18] to Hierarchical topic analysis and visual pattern mining. Novel parameters were introduced by explAIner [12] to classify XAI models [21].

We also find some existing research that sheds light on the effects of input and weight perturbations of the output on a neural network [33]. This paper also suggests an explanation to express a relation between neurons output errors and its input and weight errors.

3 Problem Characterisation

One feature common in every ML model is the enormous size. Optimizing these models further needs a complete understanding of intrinsic features. While we have methods to visualize ML models, most of them either represent only the higher-level picture and others offer detailed information but on local levels or outputs [21]. At times, these explanation methods need to be mapped to the features of a given domain, post which they are fragile to change in data, domain or model. Moreover, if an ML expert who works on multiple models from varied domain wants to get some insights about his model, the time required to implement existing visualization models makes it far more impractical to use them. Even if we consider that they are implemented in no time, all of those models work on a feature level and may overwhelm the user by presenting granular details. To create a method that can be useful for a broad application range, we can use something very generic that highlights crucial information about the model. This guides us to think of weights, as their development over time reveals a model's learning patterns which may further point to interesting insights. Even though weights looks promising to proceed with, it comes with complexities of huge data size and number of variables.

The fact that values are both positive and negative makes it complex to apply methods like binning due to possibility of oppositely signed values neutralizing each other. While most of the existing research is directed towards external features to explain the model, there is also a possibility of deriving insights from intrinsic features. Sighting the research gap between the aforementioned methods, we proceed with our research towards exploring the solutions towards possible usability of fundamental features.

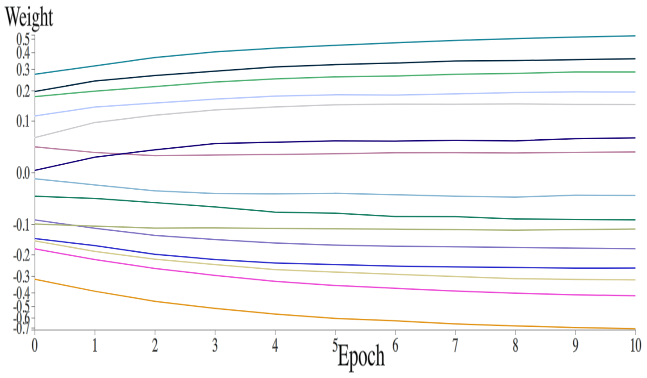
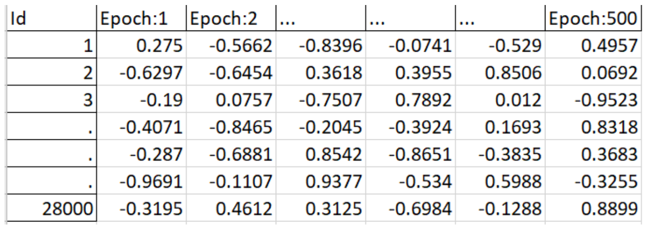


Fig. 2. A combination of log scale and interactive brush applied on figure 1. This directs the focus towards specific epochs and present detailed view by assigning more space to weights near zero

4 Choosing Feature to Explain

We begin by explaining that the target here is to make the user get insights about the ML model that can assist him to assimilate the learning and point out the possible reasons for bad accuracy. A bigger challenge is doing all this irrespective of the kind of data, domain or model. Likewise, for the same reasons, we are assuming our end user to be an expert in ML applications with a sound understanding of mathematics.

While many other researchers have tried to solve this seemingly easier problem, the crux of the problem lies in the fact that any ML model requires domain knowledge to understand the features and results. Now, when researchers try explaining these outputs, they ended up tailoring the system to fit the domain needs. This, on one side had a great advantage as that particular given domain got its explanations right and hence an output can be explained to a considerable extent while, on the other hand, the solution was useful for that domain only and will need constant updating. Some examples of such models are LIME[24] and ANCHORS[14] this make us realize the fact that despite putting efforts, the usability of any explanation gets limited to the data, model and domain that it supports. Thinking this, we intensify our research in directions of developing a solution that supports the needs of diverse domains rather then just specializing for one. However, this seemed implausible at the application level which directs our path towards the fundamentals of ML decision making and focused our attention on data, model, weights, bias, activation function, inputs layer, output layer, etc. While they all seem useful for research in XAI, we narrow our focus based on bias and weights to be most useful for the explanation as they impact the model at minuscule levels. Among these two, weights proved more informative in finding fractions of the model that were learning, segments of the model that were contribut-



Id	Epoch:1	Epoch:2	...	...	...	Epoch:500
1	0.275	-0.5662	-0.8396	-0.0741	-0.529	0.4957
2	-0.6297	-0.6454	0.3618	0.3955	0.8506	0.0692
3	-0.19	0.0757	-0.7507	0.7892	0.012	-0.9523
.	-0.4071	-0.8465	-0.2045	-0.3924	0.1693	0.8318
.	-0.287	-0.6881	0.8542	-0.8651	-0.3835	0.3683
.	-0.9691	-0.1107	0.9377	-0.534	0.5988	-0.3255
28000	-0.3195	0.4612	0.3125	-0.6984	-0.1288	0.8899

Fig. 3. A typical csv of model weights. Rows denote Ids/feature vector and columns depict how their values change over training steps/Epoch.

ing to the outputs to a higher degree along with the point in learning steps where model stops to learn. We realized that further study of weights can even help us find out the parts of the model that we can prune with least impact on accuracy. The last point seems absurd in a world where we are constantly thriving for accuracy but then there are also scenarios where an ML algorithm can not be applied only because the processing power it needs is just not feasible ,affordable or available in today's world.

5 Model, Data Set and Data Abstraction

In this section, we define the progress from choosing features to design through an iterative pipeline. Starting with selecting data set for analysis during research and checking appropriate models. Followed by analysing features and statistics for data abstraction.

Before proceeding to visualization we would like to introduce the data sets we use in our analysis. These constitute MNIST[3], Cifar10[4] and Twitter airlines review data sets[2] with CNN and dense models for analysis. As we progressed through various methods for visualisations , we analysed each of them on data sets and models mentioned above. For the sake of comparison, all the figures mentioned till section 6.5 are created using same input i.e MNIST data set and Dense model. This model demonstrated a sharp increase in accuracy for initial epochs (Detailed statistics about accuracy and loss are shared in appendix).

We then develop a python script to print weights during the training process and store them in a .csv file. These csv file store 2 dimension data (Training step, n dimension feature vector) of weight distribution during learning. We also plan to check our visualization on data with a varied loss, hence we altered our models to generate results with range of loss. This includes the addition of an extra layer, changing the number of units in layers, using different activation functions and updating the learning rate. For a small ML model with 2 layers and 50 Ids in each, training on MNIST data set will have approx 40,000 individual weights on any given training step. Representing all these weights individually will choke any visualization and may render it useless, while abstracting data may result in loss of information by averaging out the weight fluctuation of individual Id in group.

Basic methods like mean and median that are easy to apply and explain, what makes the choice difficult is the trade-off between abstraction and loss of information. Moreover, weights being both positive and negative makes the grouping of data a critical operation and needs detailed analysis explained in next section.

6 Visualization Design and Strategy

For visualization design, we start with analysing complex visualization including hierarchical edge bundling, node-link tree and radial cluster layout that suit initial needs of research. However, as our study progressed towards complex methods like DTW (explained later) and clustering, we direct our focus to a relatively simple visualization. Multi-line chart suits best here for it is simple to be understood and gives the user the ability to focus on understanding the complex methods as well as the leanings that these patterns symbolize. Next, we planned on designing a strategy to make our visualization interactive, flexible and scalable. Therefore, we devise a plan of methods and methodologies that encompass our visualization. The goal of having the strategy is simply to make the user understand the distribution of weights during the training process which she/he can further use to form a hypothesis. This primarily includes:

- Interactive zoom - We are operating with complex models, It is certain to receive data which can not be presented along with the details on one screen. On the other hand, omitting a local view may abstain the user from noticing the details at granular levels in the model. We choose to deploy interactive zoom over training steps. This feature zoom in to the selected interval on y-axis using brush pattern.
- Flexibility over the number of bins - User can select the size based on the level of detail she/he is searching for.
- Model States - We are collecting weights throughout the training process and representing changes in data over training steps which makes single model state as best suited explanation strategy in our case
- Visualization scope - To build a visualization based on intrinsic features, is most important to select a model-only visualization which primarily aims at presenting the fundamentals rather than explaining outputs [27]
- Log scale - Its useful to display numerical data over a very wide range of values in a compact way by assigning more spatial space to values nearby zero.

Once through with these, the next big step is summarizing the data. To avail easy comparison with the focus on representing a large amount of information as simply as possible, we used the following aspects of summary statistics.

6.1 Central Data Tendency

This includes statistically calculating the tendency of data to cluster around a middle value [29]. In our context, this measure is used for summarizing the data to present it to the user for visual exploration. It starts with a simple logic of using the mean to represent a rather larger data set. Though it looks simple, but forms the most complex part of our project which requires an extensive research. Major reason for the same is that we want to present a wide data range using single visualization with least loss of information. For the same purpose, we start evaluating relatively easy methods and progress further.

6.1.1 Mean of all training steps

Though it gives a viable view of Central Data Tendency (CDT), but it completely blankets the visualization of the learning process as all the training steps are now combined into one. Thus rendering it an implausible option.

6.1.2 Mean of all Ids

It has an advantage of not compromising the training process visualization, yet it comes with its own limitations that includes:

- Negative and positive weighs neutralizing each other.
- All Ids being combined to one thereby hiding the lower level details. As seen in figure.4 , all the Ids are represented by single line. While both of the above mentioned problems looked dissimilar, their solutions i.e. sorting and binning (as we will study later in this section) are interconnected.

6.1.3 Mode

It can turn useful to inform the user about the most occurred value at a given training step. But, this has its own limitations :

- It tells the user most occurred item, but not the frequency of its occurrences.

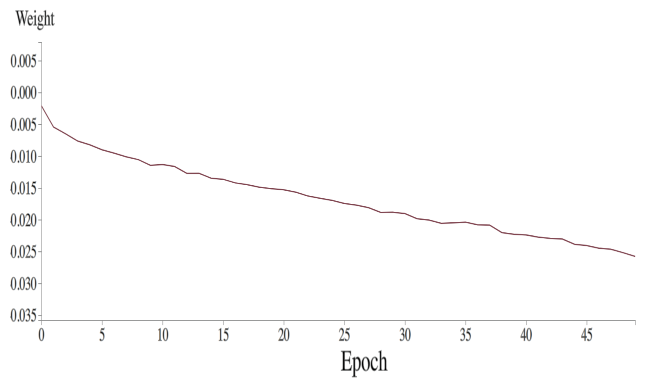


Fig. 4. A line chart representing the development of weights over training steps/epochs. Here, all Ids are represented as a single line using mean.

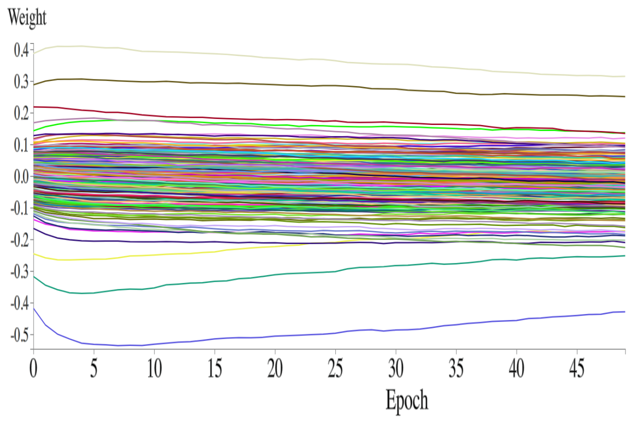


Fig. 5. Sorting weights in ascending order of values at first epoch. The straight lines are due to +ve and -ve weights averaging each other within a bin.

- It may miss outliers. On the contrary, it has one positive point over mean, i.e. there is no chance of positive and negative weights averaging each other While the above factors prove that mode may not be the best indicator of CDT, we will use mode at some later point just to add a different visualization to the user which may prove helpful in many cases.

6.2 Binning

We use binning to make bins to group the adjacent/similar weights together. This helps the user understand the intermediate picture between the mean (all data presented using a single line on a line chart) and every individual Id presented by a line (the chart becomes obsolete as soon as the number of Ids exceeds 100, which it does in most cases). As discussed already, binning possesses a huge challenge as it may put weights with different signs in the same group and they end up averaging each other out. Furthermore, in binning, we considered evaluating the following options:

Equal Width Binning - Though it easy to understand, the idea for using it is discarded as weights are highly dynamic and their range may drastically differ from one model/data to another. Fixing width/weight intervals may prove obsolete and ineffective.

Equal Depth Binning - This is also discarded from being used without much consideration because we notice that much of the weights exists within range of -0.5 to +0.5 and this binning makes bigger bins around the range boundaries, which again, may not be very helpful to understand the weight distributions.

Adaptive Binning[20]. The main purpose of this is to have dynamic X and Y-axis intervals to support any given range of data. Here we decide to bin the Ids as we study the limitation of "mean by Ids" we know exactly that we need flexibility over the number of bins size and hence over data abstraction and visualization. Moreover, flexibility in the number of bins may offer a visualization that can reveal interesting patterns as bin sizes vary with the bin count. Hence, we decide to use a scroll bar scaled from 1 to 200 to decide on the number of bins needed. Value one gives a visualization of maximum compressed (mean of all Ids) data while as user increase the number of bins, the data compression eases and one can see a more in-depth view. For instance, in a NN where we have 100 Epochs and 200 Ids, when you make bin count to 1, the chart displays single line i.e mean of 200 Ids at any given epoch. Now, when you set the same scroll bar value to 20, the number of bins increases to 20 and bin size is now 200/20 i.e 10, one line on the chart is now presenting the summarized view of 10 Ids rather than 200 which offer more clearer and less generalized visualizations that are closer to individual weights.

This also has a flaw, as the bins were just made out of adjacent Ids with no order of weights. Hence either negative and positive weights may even each other out, making the output unsubstantiated. This directs us further in the direction of our next step i.e Sorting method.

6.3 Sorting

We add this to optimize the binning over a n-dimension feature vector. The aim is to sort data in the most efficient way possible so as to solve the following problems [17]

- Sort the Ids so that least number of them even out each other due to their opposite signs
- Also take in considerations the Ids that change their sign during the learning process

Observing above constraints, we analyse sorting criteria with goal to order Ids in a way that places most similar weights together. With this, we start considering our options and came up with the following solutions [16].

6.3.1 Sorting by Epoch 0

Here, we sort data in ascending order of weight values at epoch 0. The visualization created using this sorting of the lines on the chart were not crossing each other as often (refer to fig:.5) as with no sorting at all but this is also almost random sorting because it laid its foundation on Epoch 0 and at that point the weighs are decided randomly. Counting on the fact that this sorting limits the visualization to the statistical analysis to Epoch 0 which indeed is almost random, we decide not to use this sorting for our visualization.

6.3.2 Sorting by Last Epoch

It consists of sorting data in ascending order of weights at the last training step. Compared to the previous approach, here the weights used for sorting have developed over time. On observing (Refer to fig. 6) a visualization on the same data, the lines depicted a diverging behavior symbolising clearer learning patterns.

6.3.3 Sorting by mean of all training steps

This further proved helpful as the lines stop crossing each other all together and are almost a straight line after the initial learning curve. Then in order to have a detailed view of its functioning, we tested it with lesser Ids so that we can analyse each Id and its impact on visualization and it was there when we realized two limitations of this sorting. It puts 2 Ids under one bin on the basis of their mean being almost same but it missed the fact that they have completely different weights but have the same mean because

- one Id started with weight in the middle of the range (eg - 0.5) that learnt nothing and other Id learnt from 0.2 to 0.8
- One Id started with negative and grew to positive and other Ids just developed other way.

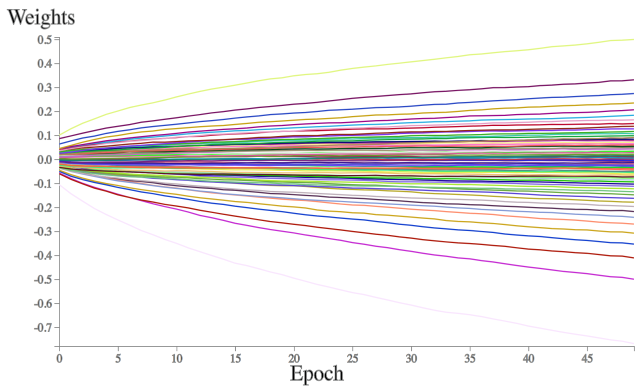


Fig. 6. Sorting weights in ascending order of values at last epoch. As compared the First Epoch sort, the lines are diverging, reflecting development of weights over time.

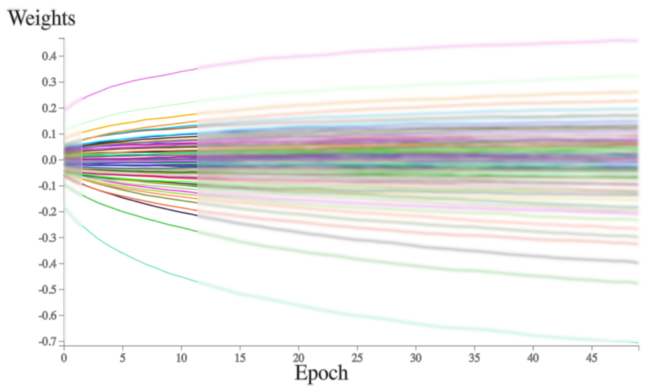


Fig. 7. Sorting weights in ascending order of sum of Weights for individual Id. Focused area reflect the more fluctuating learning patterns as compared to Last Epoch sort in figure. 6 .

6.3.4 Sorting by Standard Deviation

SD (Standard Deviation) is a significant method to measure learning on a statistical scale. This method is vital for further optimization of the model as well as for pruning. Assuming Ids having similar standard deviation are learning at the same rate, we sorted data based on ascending order of SD for each Id. However, we have to tailor SD because its standard formula calculates positive SD for any given id, sorting by which may put two feature vectors with almost same values and different sign adjacent, when this data is binned, they will even out each other. To tackle this problem, we have added a sign of last training step to standard SD output. With the modified formula, a user can determine the degree to which a model is learning by looking at the SD visualization for any given model. We realized that even though SD is significant, using it to present all weights doesn't solves our objective. Hence, we discarded this approach but took a note to use it at later stages along with filter feature.

6.3.5 Sorting by Sum of Weights

Sighting the limitations of SD and mean, we developed new sorting criteria. Here, we sum all the values of Id over epochs and then sort features vectors in by ascending order of this sum. The features meet what was left as a limitation by mean and SD sort. To test it further we created a visualizations with same data. The resultant figure.7 illustrates more curved learning patterns in initial epochs as compared to ”Last Epoch Sort”. We inferred this as an important finding and finalize this method for our final visualization method. One limitation that comes along with this method is it can still put weights that change to the same degree and have the same values in the same bins, despite considering the possibility of them having exactly opposite signs. For instance (0.5,0.2,0,-0,7) and (-0.7,0,0.5,0.2) .

6.4 Dynamic Time Warping and Agglomerative Clustering

Having evaluated various methods for grouping, we further decided to proceed with the check usability of time series methods [7] for our project.

Here we focused on classifications of time series analysis methods and their representation methods. We also analysed that in our scenario, similar shape cluster can be effective to represent learning. For the same reason, we analysed algorithms where the time of occurrence of patterns is not important to find similar time-series in shape[22]. Some domains already work to find such clusters of time-series with similar patterns of change are constructed regardless of time points. For example, to cluster share price related to different companies which have a common pattern in their stock independent on its occurrence in time-series [7] [31].

Due to its efficiency in dealing with temporal drift and shape averaging, we decided to evaluate the usability of DTW (Dynamic Time Warping). This method seemed promising, given that all the methods we study in the last section can only group data on overall values

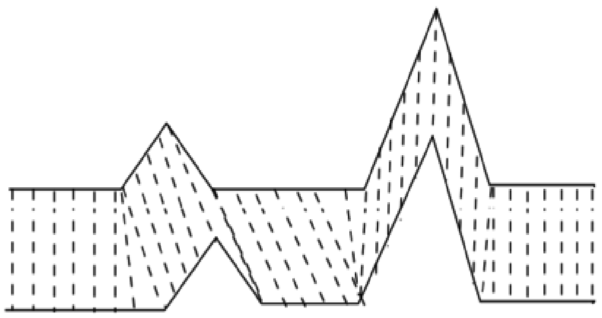


Fig. 8. Optimal map in Dynamic time warping, it allows the algorithm to find common learning patterns irrespective of point in time [1]

of given Id over learning steps or just group them in sorting order of some training step. What they all have in common is that none of them takes into account the change in values during learning. Furthermore, with DTW we will get an opportunity to consolidate the learning patterns that we want to visualize, with its feature of the optical match i.e. comparing weights across the time steps which means if we have two Ids with same learning rate but different time steps for learning, Using DTW we can measure the optimal similarity between these two, as shown in figure 8.

Having calculated the distances of all epochs, we now have to group them. For this, clustering seemed promising because there is not any early knowledge about classes. we considered hierarchical clustering approach of unsupervised learning where each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy offering complete flexibility over the number of bins.

Once clusters are formed, we worked on appropriate visualization to present a given cluster for visual exploration. Now to represent with multiple records within a cluster we look in direction of CDT. We also take a look on other possible methods or concepts that could aid a more detailed perspective of model (Refer to figure .10). we decided to use KNIME as a visualization tool to evaluate few methods on clusters. The idea behind using a diversity of methods is to facilitate user with disparate visualizations to offer the possibility vivid perspective for the same data. For eg. while mean will represent the average weight, mode could be interesting when the user wants to know about the most common weight in a cluster.

6.4.1 Mean

The most basic form for data summarizing and most susceptible to outliers, this quality of mean makes it useful to detect outliers weights.

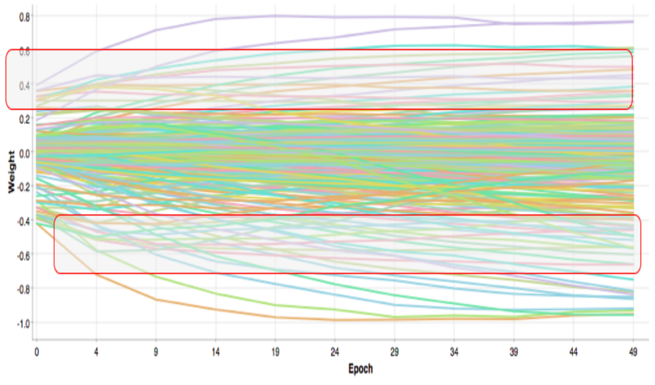


Fig. 9. A combination of DTW, Agglomerative clustering and mean on weight data. Highlighted in red are the segments where weights show more fluctuations as compared to Weight Sum Sort in figure. 7

As shown in figure. 9 , the lines are more variation for same data, also there are fewer lines overlapping each other as compared to any prior visualization.

6.4.2 Mode

While it represents the most common weights amongst the bin, this offers us patterns representing more individual fluctuations in weights rather then averaging them.

6.4.3 Median Absolute Deviation

We already ruled out using the median as it loses the frequency information, how about finding the deviation from median ? as it may help us find the diversion of data around a central point, this leads us to mean absolute deviation that comes from the area of statistical dispersion, where dispersion is (also called variability, scatter, or spread) the extent to which a distribution is stretched or squeezed. Mean absolute deviation is a common method to measure the variability of a univariate sample of quantitative data.

6.4.4 Regression, DTW and Agglomerative clustering

Undoubtedly, the combination of DTW and agglomerative clustering proved to be of great use for our research but never the less, there is space for improvement. considering same, we tried to juxtapose one typical method used in machine learning i.e. linear regression (as shown in figure.10). In general, it's used as a method for ML but in this context, we are using it to further refine clusters by dividing the weights into 2 clusters based on their linear regression coefficient. We start with applying DTW on weights, the next step is to calculate the regression coefficient of all training steps for each id. Once done, we divide the weights in a group of 2 based on the sign of the regression coefficient. Post this we apply agglomerative clustering on each of the group. Once clustered, we visualized them individually as well as in combination.

As seen in figure.11 the lines reflect more sharper fluctuations in learning patterns.

6.4.5 Breaking visualizations from model to Layers

Having worked multiple approaches to abstract the data. we recognised that the visualization can be more interesting and useful if we break it down to even lower levels of model i.e layers. This may be specifically useful for models where the number of Ids witnesses a drastic change from one layer to another as in that case the common weight of Ids is drastically different and presenting all on the chart comes with a trade-off for details. To achieve the same, we further altered the core of Tensorflow to yield layer-specific weights. The visualization didn't just present with details as the chart was not formed to fit the range of just one layer's weights and the patterns which were otherwise suppressed to minuscule levels earlier now emerged more clearly and offered an even better data exploration.

6.5 Location tendency

This determines the "location" or shift of the distribution. We have used the concept of quantile to subdivide the weights of distribution into equal proportions. While the idea occurred naturally to sort data by our defined sorting methods and then divide them in quantiles but that eventually would have just been visualization to the statistical analysis of sorted Ids. While in Location tendency, what matters is the value of data present at any given training step so we can completely forget Ids and sort data on every Epoch step to assign it to related quantile. The objective of this feature is to inform the user about the distribution of weight nearby zero, insights about this will further form base information for the possibility of model pruning (refer to figure 12). We sailed through multiple approaches before finalizing, these were as follows First Training Step Sort, Last Training Step Sort, Weight Sum Sort, Standard deviation sort and Individual training step sort. While all the above sorting methods claim to have some merits over the others, one common pattern followed by all is that they sort the data once and for all. The approach we used to calculate quantities is useful to analyse the overall weight distribution and also can be considered as a factor to determine the quality of model but can not be used as individual factors to get insights for pruning as the Ids are not taken into considering for this Viz.

7 Utilizing Visualizations

"Visualization gives you answers to questions you didn't know you had." – Ben Schneiderman This phrase exactly fits the scenario. We are trying to present data without having a complete idea of the use case that will be beneficial to the user. The overall goal is to present information in a balanced combination of high-level overview and detailed insights at the lower level.

Studying from a higher perspective, the most common reasons for bad accuracy [11] [13] are

- Not enough training data
- Inappropriate activation function
- Too high or too low learning rate
- Inadequate training steps For a perfect visualization shall assist a user in ruling out some of the above possibilities if not point the exact one of the cause for a bad model.

To achieve the same, the visualization we developed offers the following features

1. Visualize the timeline overview of the learning process of a model
2. Can understand the central data tendency
3. Can avail "on-demand insights"
4. Understand the distribution of weights
5. Comprehend the spread of weights over time
6. Perceive the range of weights over given training step

Along with analysing the cause for bad accuracy, the above steps are the measures, if applied over a model with good accuracy, that may assist the user to the conclude whether or not to prune the weights and if yes, which and how many weights to prune with having least impact on accuracy.

8 Use Cases

Here we present two cases to show how the workflow is represented in our system. First use case explicates the use of visualization for pruning while the other one facilities a developer who wants to enhance the efficiency of the model

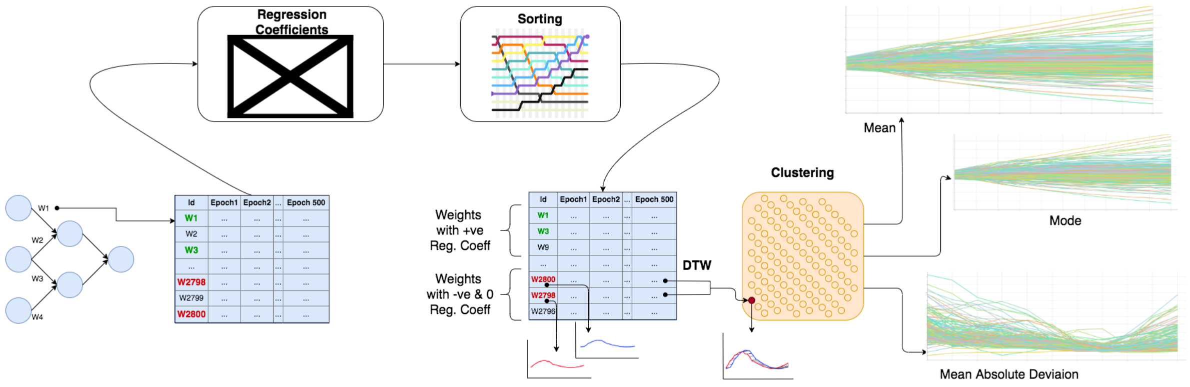


Fig. 10. Flow chart explaining methods used in combination with DTW and agglomerative clustering.

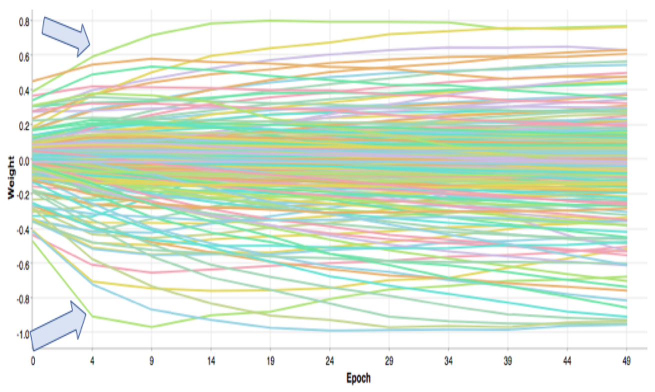


Fig. 11. Regression grouping, DTW and Agglomerative clustering applied to weight data. The arrows point to area which is more spatially distributed as compared to visualization without Regression 9

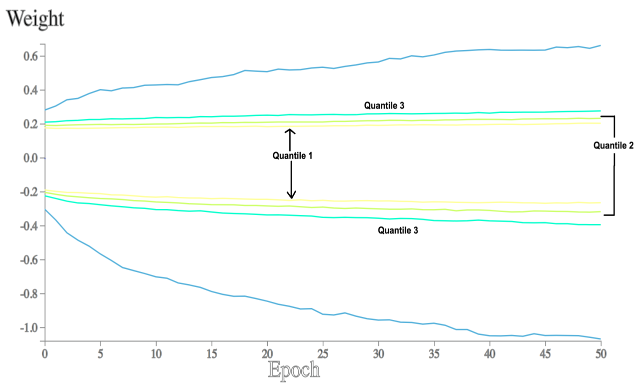


Fig. 12. Lines showing the quantile division of weights in a model.

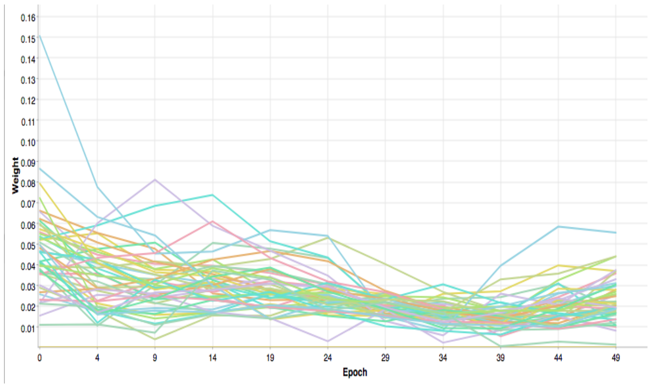


Fig. 13. Lines presenting SD of clusters formed by combination of DTW and agglomerative clustering.

8.1 Case 1

The first use case covers examining the model for dead weighs. In this scenario, the user is running a three layered dense model on MNIST[3] data set. He wants to reduce the processing time of the model by pruning dead weights. To locate these weights, she/he analyses our method to comprehend location, distribution and learning pattern of weights . Here is how she/he utilizes the available visualizations to assist his decision making.

1. Mean - This can give an initial estimate of the number of weights that are not learning.

2. Location Tendency - With quantile visualization (figure 12), the user registers that 3 quarters of weights lie in the interval of +0.2 to -0.2. She/He also learns that quantile 2 and quantile 3 lie in very close range of quantile 1.

3. Standard Deviation -Based on the above insights, the user decides to filter set Sd filter criteria to 50 percent. This excludes 50 percent weights with least SD and presents the remaining weights.

On observing the SD chart (figure .13) the user understands the learning degree of each cluster formed by DTW and agglomerative clustering. Relying upon his learning from visualizations, she/he decided the prune 25 a quarter of weights with least SD from the network.

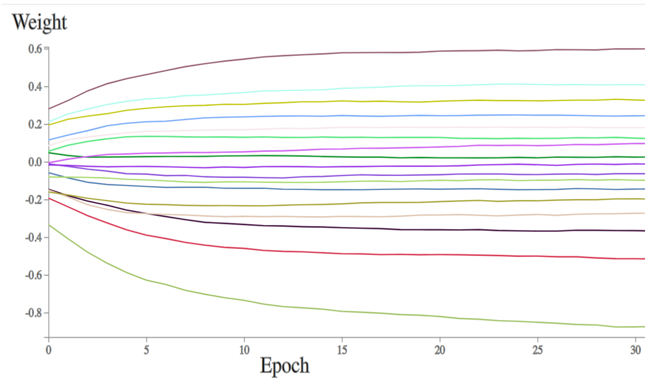


Fig. 14. Weight Sum visualization of a ML model. The lines diverge initially but grows almost straight in end.

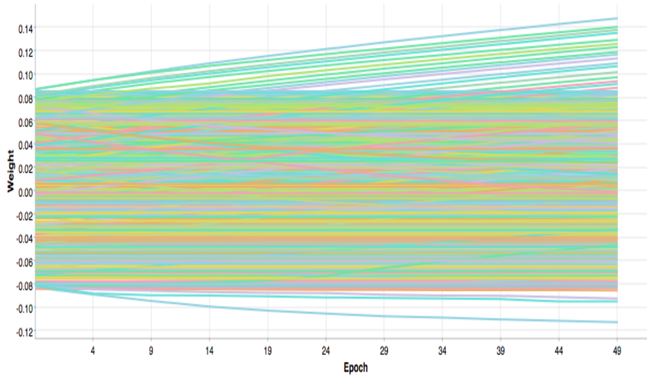


Fig. 16. Visualization of first layer weights of a ML model. The straight lines represent absence of learning in later learning steps.

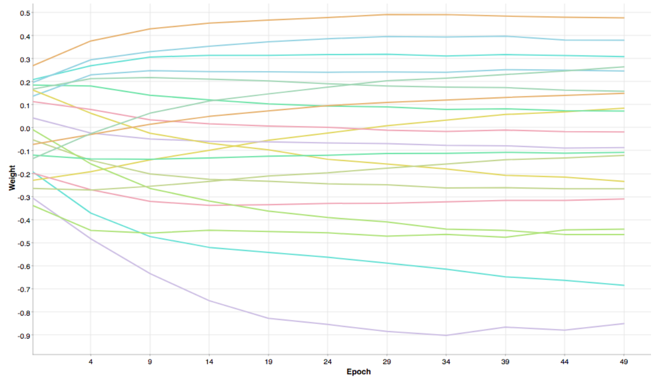


Fig. 15. DTW and agglomerative clustering over weights for initial layer of a dense ML model.

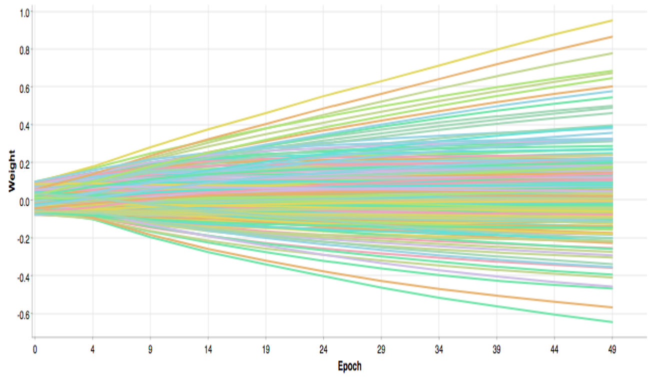


Fig. 17. Visualization of first later of same model as figure.16. This model is now trained on additional data.

8.2 Case 2

A physicist(AI expert) develops a ML model to classify the particle images from the microscope. On the first run, she/he gets extremely low accuracy and validation accuracy. Our visualizations assist the user in exploring the insights further. Visualizing the weight sum sorting for the model (refer to the Fig. 14), she/he infers 2 facts about the model i.e.

- Weights are diverging initially but are almost stable by the last training step
- The lines at the bottom of chart is showing different behaviour then rest of lines

While the first point rules out the possibility of a shortage of data or epochs the user suspects that second fact can be indicative of some crazy weights which are impacting the accuracy. To understand further, she/he decides to visualize the individual layer with DTW (Refer to fig 11) , agglomerative clustering and mode. After noticing that everything seems normal in all visualization, she/he concludes that inadequate data or training steps are not the reason for this accuracy and she/he then continues his analysis further to check other reasons on his own.

8.3 Case 3

An AI expert at airlines company runs an AI model to predict the seat occupancy for flights. The idea is to predict the seat occupancy using data of the first quarter of 2020 and based on that decide, the flight frequency to various destinations in upcoming months. Compare to the results of the same model last year, the user witness a sharp decline in validation accuracy while accuracy is almost intact.Using our viz the user first observes DTW and agglomerative visualization (refer to figure. 16) for the model where she/he visualizes most of the lines to be almost straights which completely resembles less learning. Something that caught users attention is that there are some fluctuations in weights in initial epochs. Which clearly reflects the following facts about model

- The training steps are optimal
- The model finds perfect weights which represent train data and this happened soon after the start

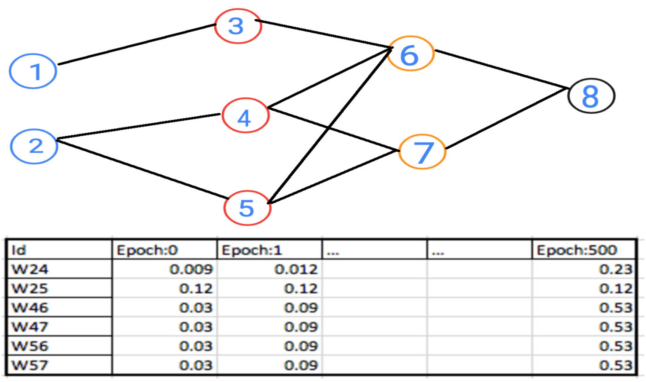
To further understand the model she/he views the DTW visualization for other layers which present almost same pattern.

Concluding the above steps along with the simple loss landscape, the user decides to add more data to the training set and rerun the model. Now the model matches the validation accuracy expectations signing which the user wants again to visualize the new model to understand the cause of this change on visual grounds for which she/he sees a chart(figure. 17) that has barely any straight lines.

9 Opportunities, Challenges and Discussions

In this design study we researched various methods to study and analyse weights of ML models while we were able to come up with a useful visualization, but at this point, we have understood in detail the concepts of dynamic time analysis, limitations in binning of weights distribution, and have acquired knowledge about mathematical concepts that may be useful in this area. Our research also opens some gates to the following new opportunities for future work.

- Integrating our viz tool with Tensorboard - Tensorboard is a commonly used tool to visualize for machine learning experimentation. Integrating our visualization as a plugin to Tensorboard will



Id	Epoch:0	Epoch:1	...	...	Epoch:500
W24	0.009	0.012			0.23
W25	0.12	0.12			0.12
W46	0.03	0.09			0.53
W47	0.03	0.09			0.53
W56	0.03	0.09			0.53
W57	0.03	0.09			0.53

Fig. 18. Weights 46,47,48,49 will contribute equally to visualizations despite the variation in learning of their preceding Ids.

make it handy to use. Hence, can benefit many ML experts with vital statistics about their models

- Allowing changes in the model through GUI - Currently, user can only analyze the weight in our visualization. One important use case of this visualization is when a user uses this data to prune the model. An additional option to allow the user to prune the selected weights right away from GUI will thereby save the efforts of going to code and remove the weights manually. Further, rerun option may prove even more useful as a user can just perform everything in ”exploration loop” [25] right from the GUI from analysing weights, pruning, rerunning and analysing the impact due to his changes.

- Utilizing the layer hierarchy - Data records often have some relationship to other pieces of information [19]. Right now we apply operations to weight without considering the impact of the weight of antecedent neurons. We can use this fact to further refine the process of binning. Let’s take an example of the model as shown in fig:15, In the given case, the weights 46,47 and 56,57 appears to be have same values and learning at the same rate. All four will be considered the same for all our visualizations. While we completely ignore their dependence on preceding connections. In this case weight, 24 which forms in only connection of 46,47 is learning far more than 25 which indeed is responsible for weight 56,57. The idea here is to research these relations to seek possible impact over upcoming weights and this may prove an even better factor in binning the weights.

During our research, we came across a question about the domain which we though to propose here in this section. We focused to produce a viz that is independent on all three factors determining a NN i.e Data, Domain and Model. Though we were able to achieve a platform that helps the user understand about weights and decide about the quality of the model. Many times during this time, we felt of using some concepts that completely explained the problem at hand but when we applied the same to other models/data sets, it made no sense at all. This left us with very limited options to choose from. While making those trade offs, we felt, if is it really plausible to think of a model that servers all ML models, data and domains or this will end up with insights too assorted and generic that user may not find it enough interesting.

10 Conclusion

We started with analysing the existing XAI methods and their dependencies. Post that, we ponder upon intrinsic features of ML models which we can use as a foundation for XAI method. This was followed by presenting ways to sort and bin data. Research also lead us to development of new criteria for sorting i.e. weight sum sort which proved a step above traditional sorting methods. Apart from standard methods, we introduce the utility of Dynamic time warping, agglomerative clustering and their combination with regression for analysing patterns of learning. Besides the methods, we also customize the tensor flow to produce weight files for individual layers which provided us with a more detailed perspective. We used Knime and D3 to create a visualization that offers the user control over bin count, clustering method and sorting method. The interactive zoom and quantiles were designed specifically to make sure that the user has both upper and lower level views available. Use cases presented the description of how our method can be used to determine the quality of machine learning model in a meaningful way that empowers an ML expert to run plug and play analysis for the model without having to go through time taking configuration process. At last, we mentioned the questions that we couldn’t answer and carries potential to prove a big leap in direction generalizing XAI.

References

[1] Dyanmic Time Warping. `https://en.wikipedia.org/wiki/Dynamic_time_warping`, 2015. [Online; accessed 19-July-2019].
[2] Twitter US Airline Sentiment. `https://www.kaggle.com/crowdflower/twitter-airline-sentiment`, 2015. [Online; accessed 19-July-2019].
[3] Tensorflow Data Sets. `https://www.tensorflow.org/datasets/catalog/mnist`, 2018. [Online; accessed 19-July-2019].
[4] Tensorflow Data Sets. `https://www.tensorflow.org/datasets/catalog/cifar10`, 2018. [Online; accessed 19-Feb-2020].
[5] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.
[6] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.
[7] S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah. Time-series clustering–a decade review. *Information Systems*, 53:16–38, 2015.
[8] S. Amershi, M. Chickering, S. M. Drucker, B. Lee, P. Simard, and J. Suh. Modeltracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 337–346, 2015.
[9] A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks*, pages 63–71. Springer, 2016.
[10] M. G. Core, H. C. Lane, M. Van Lent, D. Gomboc, S. Solomon, and M. Rosenberg. Building explainable artificial intelligence systems. In *AAAI*, pages 1766–1773, 2006.
[11] R. H. DAVIS, D. Edelman, and A. Gammerman. Machine-learning algorithms for credit-card applications. *IMA Journal of Management Mathematics*, 4(1):43–51, 1992.
[12] M. El-Assady, W. Jentner, R. Kehlbeck, U. Schlegel, R. Sevastjanova, F. Sperrle, T. Spinner, and D. Keim. Towards xai: Structuring the processes of explanations.
[13] S. García, A. Fernández, J. Luengo, and F. Herrera. A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Computing*, 13(10):959, 2009.
[14] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93, 2018.
[15] D. Gunning and D. W. Aha. Darpa’s explainable artificial intelligence program. *AI Magazine*, 40(2):44–58, 2019.
[16] M. He, X. Wu, S. Q. Zheng, and B. Englert. Optimal sorting algorithms for a simplified 2d array with reconfigurable pipelined bus system. *IEEE Transactions on Parallel and Distributed Systems*, 21(3):303–312, 2009.
[17] T. Howley, M. G. Madden, M.-L. O’Connell, and A. G. Ryder. The effect of principal component analysis on machine learning accuracy with high dimensional spectral data. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 209–222. Springer, 2005.
[18] L. Jiang, S. Liu, and C. Chen. Recent research advances on interactive machine learning. *Journal of Visualization*, 22(2):401–417, 2019.

[19] D. A. Keim. Information visualization and visual data mining. *IEEE transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.

[20] W. K. Leow and R. Li. The analysis and applications of adaptive-binning color histograms. *Computer Vision and Image Understanding*, 94(1-3):67–91, 2004.

[21] T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[22] M. Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[23] S. Pasricha. explainable artificial intelligence. University konstanz, 2019.

[24] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.

[25] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim. Knowledge generation model for visual analytics. *IEEE transactions on visualization and computer graphics*, 20(12):1604–1613, 2014.

[26] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE transactions on visualization and computer graphics*, 18(12):2431–2440, 2012.

[27] R. Sevastjanova, F. Beck, B. Ell, C. Turkay, R. Henkin, M. Butt, D. A. Keim, and M. El-Assady. Going beyond visualization: Verbalization as complementary medium to explain machine learning models. In *Workshop on Visualization for AI Explainability at IEEE VIS*, 2018.

[28] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[29] T. P. Theruvath, J. A. Jones, and J. S. Ikonomidis. Matrix metalloproteinases and descending aortic aneurysms: parity, disparity, and switch. *Journal of cardiac surgery*, 27(1):81–90, 2012.

[30] E. Tjoa and C. Guan. A survey on explainable artificial intelligence (xai): Towards medical xai. *arXiv preprint arXiv:1907.07374*, 2019.

[31] G.-J. Wang, C. Xie, F. Han, and B. Sun. Similarity measure and topology evolution of foreign exchange markets using dynamic time warping method: Evidence from minimal spanning tree. *Physica A: Statistical Mechanics and its Applications*, 391(16):4136–4146, 2012.

[32] K. Wongsuphasawat, D. Smilkov, J. Wexler, J. Wilson, D. Mané, D. Fritz, D. Krishnan, F. B. Viégas, and M. Wattenberg. Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):1–12, Jan 2018.

[33] X. Zeng and D. S. Yeung. Sensitivity analysis of multilayer perceptron to input and weight perturbations. *IEEE Transactions on Neural Networks*, 12(6):1358–1366, 2001.