Recording the Lifecycle of Software Models

Brendan P. Bonner
School of Computing
Dublin City University
Dublin, Ireland
brendan.bonner2@mail.dcu.ie

***Abstract*—Artificial Intelligence is built on trustworthiness of the systems that provide the intelligence. The bedrock of these are models that have been trained in a particular manner to generate outcomes that replicate, or improve, the training inputs. To ensure that models used by AI practitioners are verifiable and traceable, we propose to investigate if we can validate models generated by machine learning systems, and generate a trace ability path between deployed models and the initial training, or pretrained models if transfer learned.**

***Index Terms*—Artificial Intelligence, Provenance, Explainability, Model Analysis**

I. Introduction

The topic covers the recording of CNN layer metrics during the training sequence. The hypothesis is to examine trust mechanisms for explaining AI, and providing an immutable record of the training sequences prior to a model being utilised or customised.

The practicum focuses on two approaches to this; recording and visualisation of the sequence of changes during the training of a model. During model fitting at configurable intervals, layer characteristics are measured and recorded, alongside mathematical verification, and stored. This can be visualised, supporting modern explainable AI measures, to verify model behaviour according to training, and can be trusted.

The proposal builds upon the attempts to provide additional metrics to support the establishment of the concept of fairness in AI. In comparison with human trust, which is established through verification and measurement of academic and professional credentials, Fairness in AI is an addendum to the model being examined, with attempts to quantify via measures such as Thiel scores for certain bias tests. This is further developed by IBMs AI360 Fairness toolkit paper which show how this is measured and corrected, but not being able to identify where that behaviour was acquired and why a model is trained in a certain way to develop these outcomes.

There is currently a gap in identifying if there is a provenance of trust when using deployed AI systems, and if sibling and inherited models retain these behaviours and if this can be visualised. While attempts to visualise the internals of decisions in the Simonyon and Bau papers, these focus on trying to visualise the pathways being activated in a 2 dimensional map. The dissection and visualisation papers provide a key how to measure and record this aspect of models, and our research will examine if there is a potential to securely record neural network snapshots.

The question being asked is to determine if we develop a novel mechanism and metric for representing the evolution of knowledge within a neural network during the training and development stages of creating a model. By analysing the state of the art in terms of AI fairness and methods of visualising the dissected layers common networks, the research will establish if creating an immutable chain of layer delta changes over time will provide an insight into the trustworthiness of the underlying model.

The second part of the research will ask if the information being stored can be visualised based on the elements of explainable AI to be able to identify which iteration and epoch of the training process is responsible for the manifestation of the behaviour. Ultimately, the outcome of the research should ask if providing verifiable additional data on how an artificial intelligence system evolves over time, if this can provide a foundation into further research towards comparisons of human trust and artificial intelligence trust.

A. Synopsis

To deliver the model synopsis, a reduction algorithm was applied to each layer in the model, reducing the weights and biases to a pair of unique identifiers containing the *standard deviation* and the *variance*, representing the distribution skew of the data away from a normal distribution. Once this synopsis is applied, we were able to reduce all default keras models from several megabytes to several kb arrays that are comparabile and the basis for generation of a unique signature.

B. Objectives

The overall objectives of the practicum are as follows:

- Discover a method to extract a unique signature from any Deep Neural Network software model.
- Create a local and global verification trail for
  - A Path of provenance to all ancestor models
  - A method to evaluate divergence of the child model from the parent
- Validation of Provenance via verification and illustration

Once the signature is generated, we are able to create a immutable link to a previous model, and store the deviation of both the structure and the weights as a factor. The key element of this is the generation of a lineage for both training and validating that there is a history of the model's evolution.

II. Background

Models in CNNs are the primary reason for opacity of neural networks. Decision trees and algortihms have the benefit of being easily repeatable outside of the system. The complexity of even the simplest models makes this unviable. The introduction usually describes the background of the project with brief information on general knowledge of the subject. This sets the scene by stating the problem being tackled and what the aims of the project are.

A. The Role of Models in CNNs

A model in deep neural networks is the basis for the fuzzy nature of a certain outcome based on an input. As the input changes, the impact on weights, balances and activation functions in trained models can provide significant changes in outcomes, whether we a are classifying the image, identifying components or using the outcome as the basis for generating an outcome. In frameworks like Keras and Pytorch, the Model contains roughly the same structure, based on an input layer, a number of hidden layers, and finally a clasification or output layer. As a model is trained, the input layer and all not weighted layers remain the same, but the back propagation function continuously changes the weights and biases across multiple layers with diverse interlinks.

B. Identifying Differences in Models

Each layer is made up of a structure containing a dictionary of details, plus a multi dimensional array containing the weights for each sublayer that covers all the dimensions, plus a series of biases. While trying to understanding the internals of layers is interesting but relatively futile, being able to break down the structure of the layer into a unique single identifier that is non-repeatable across the smallest changes, but also allows comparison between instances of the layers when trained.

C. Weights and Biases

D. Literary Review

The practicum investigates the need and demand for evaluating the development of a methodology for efficiently recording the learning process of an AI model, focusing on convolutional neural networks. The vision will establish if existing CNN explainability methods can be used to define a novel temporal chain of major differences in the underlying structure of the network during training.

Once we have determined what elements of value in the process show what can be recorded, it will create a verification repository for the network (similar to a CV) and show how this can be used as either a trust chain when for establishing the provenance of the model, reviewing the network training in comparison with the post-network visualisation, or providing a mechanism to see if the training process is either optimal or biased.

The research area overlaps between the current drive to explain black box models to address trust issues with artificial intelligence, with the drive to be able to optimise and evaluate the process for creating and expanding existing models. The bulk of current research focuses on the behaviour of neural networks after training, examining how an input triggers various layers, and this has reduced as deeper networks become more commonplace. Starting to record and validate how networks are trained, rather than used may provide additional support to trustworthy AI, while benefiting training evaluations and visualisation of when neurons are activated during the learning process.

a) State of the Art: What is the state of the art in relation to measuring the layers and understanding convolutional neural networks for explainability.

- Overview of characteristics of the theme (commonalities, differences, nuances) In the process of understanding and trusting deep neural networks, researchers have been towards extensive evaluation to identify pathway activation, or by presenting input triggers and looking at activations between layers. There is a difference in approach regarding what has generated use on explainability, allowing multiple approaches to cover the effect of individual features, morphology characteristics or how pixels resonate various components of the network. The focus is to understand the network, rather than understand how it is trained. The closest research in understanding the training process is the use of classifier probes, and how these can be recorded in a provenance chain.
- Examination
  - The first step in our journey to evaluating the trustworthiness of AI systems is examining how fairness and bias is interpreted across all artificial intelligence systems. In AI fairness: how to measure and reduce unwanted bias in machine learning, Mahoney, et al, examine each phase of the AI deployment process to identify where bias is created, identified and countered. The resultant research created the IBM AI Fairness (AIF360) toolkit [1], which looks at all network types and rewires the network weights and biases to remove predefined bias from the outcome. This works well on business AI systems classifiers like random forest and naive bayes, and simple neural networks, but is highly limited in deep neural networks. The research does identify one important gap - there is no mechanism to alter a trained network, and expect it to behave as before. Part of the research will provide a prevalence trace to verify that the network being examined is mathematically verified to not be tampered with, potentially increasing the trust in deployments.
  - While the above paper looked at basic networks, the journey is more difficult for convolutional neural networks due to the increasing depth of networks. A popular paper on understanding the decision process within ConvNets was discussed in the 2014 paper Deep Inside Convolutional Networks [2] where the innards of a CNN when presented with an image

to be classified was visualised using saliency maps. The method of saliency extraction with a view to presenting a single 2D image map gave a "last layer" backwards view of the network activations, as if you were looking through a telescope, while imagining the neurons. This research highlighted that aggregation of individual neuron activations can be utilised for explaining CNNs. What we would like to see is how to twist this view around, and instead of seeing the activations of neurons, instead see the neurons being trained.

- The final piece in the jigsaw is attempting to use this understanding to make a score of interpretability and the experiments of breaking down the components of CNNs are discussed in Network Dissection: Quantifying Interpretability of Deep Visual Representations. In this 2017 paper, Bau et al, introduces the concept of scoring unit interpretability to give a segmentation threshold to an individual concept when the CNN is classifying. The attempt is to visualise and score diverse CNNs to evaluate discrimination and interpretability. The outcome of this research will be to examine if there is a link between the number of detectors of concepts in the outcome tests to the number of times a certain portion of the network is trained to classify or ignore that concept.

b) Interpretation of the Learning Process:

1) Overview of the Theme Once we understand that there is a gap in being able to visualise the learning process when training each convolutional neural pathway in an AI system, we then have to examine the state of the art in being to interpret this training process. In this part, research is quite light, as the focus has been on understanding a network when it is making a classification or using evaluation to interpret differences in classification. The approach that I will take is to examine how individual elements of each layer are tweaked during the training process. While a brute force approach would be to record the loss function changes in back-propagation for every trained step, this would result in a compute requirement of 12m parameter changes each step, which would require multi-terabyte storage for even a competition trained VGG-16 network. The approach has to be optimal to record only changes in neurons that are beyond an average activation threshold, and then only in a constant dimension depth per layer. The resulting delta will be configured everyX steps or epochs to provide a consistent path of evolution during training.
2) Examination
   a) In assessing the Understanding intermediate layers using linear classifier probes [3] , we finally get an appreciation of the value of examining the learning process as opposed to the post-trained model activations. Although still concerned with feature activation, it introduces a new method that sits in parallel to the traditional neural network library of tools. In this case, monitors of features at every layer of a model are evaluates suitability for classification. The approach is to have completely independent linear classifiers, which is needed for delivery of this theme. This was applied to two large models in Inception v3 and Resnet-50, and the resulting analysis of similar networks helped us focus on high performing and large networks like VGG-16. The issue with this research in line with the theme is the focus is on interoperability, the key outcome is how linear separability of features change throughout the model, although it provides the best research available to optimise the training for visualisation.
   b) A further exploration at examination of the internals of a CNN, and the introduction of identifying how low level concepts interact with high level features is put forward as a proposal for Quantitative Testing with Concept Activation Vectors. In this 2018 paper, Been Kim, et al, ,agree that explainability remains a significant challenge in the AI world, but is still important to attempt to de-opaque to make sure that a value based machine learning world is possible. While reference is paid to Alain and Bengio's previous 2016 findings, this goes beyond and uses the CAVs for interpretability for testing. In particular, we can rework the proposed extensions to TCAVs by seeing if we can establish if the learning pattern grows with the shape, texture of repetition of the training images. The findings were also used to develop artistic style generative images based on the test outcomes. While we will not attempt to replicate the Empirical Deepdream, visualisation of the outcomes, similar to the Simonyon outcomes above should be available.

c) Visualisation of Convolutional Network Values: Focus on the exploration of data layers of a CNN - Explore the layers of a neural network with your camera and how this evolved from GradCam. More traditional visualisation techniques are needed to better understand CNNs, and we will use the same approach. More complex 3D imaging will be required due to the need to see temporal differences in the network over time.

1) A close examination of visualisation techniques push back to the development Of the open Stuttgart Neural Network Simulator (SNNS), which allowed the early definition and visualisation (based on X11 graphics) to get the size and scale of the demostrations of the largest A good example of research is Adam Harley's visualisation tool [4] that shows the full CNN complete with all interconnections, and weighted values that are activated on (in this case MNIST) data and a hand drawn input. DOI: 10.1007/978-3-319-27857-5 77 This

is based on previous work, such as the stuttgart Neural Network Simulator (SNNS)

2) There is also some good visualisation of the network by using the ConvNetJS library, which looks at training data for activations, but again this is based on visualisation of the network and how the weights impact the next layer. This network has 1024 nodes on the bottom layer (corresponding to pixels), six 5x5 (stride 1) convolutional filters in the first hidden layer, followed by sixteen 5x5 (stride 1) convolutional filters in the second hidden layer, then three fully-connected layers, with 120 nodes in the first, 100 nodes in the second, and 10 nodes in the third. The convolutional layers are each followed by downsampling layer that does 2x2 max pooling (with stride 2).

d) Overview of State of the Art: While there has been continuous research and improvement into the field of explainability, there remains a disconnect between the understanding of why a neural network produces a specific outcome – of which research is plentiful; and a lack of investigation into establishing a trust quotient to provide verifiable classifications based on either its initial training or amendments provided after transfer.

The benefit of the research is there is a clear metric and methodology to verify what components of a CNN are activated by a source image, and we can do this not only in an interesting way that can show how far down a deep network the feature or concept is important, but also to guide optimisations. CNN visualisation is also strong, but again based on the use not how it was trained.

What is lacking is a guidance as to why the training of the network led to this behaviour. If we create an analogy of a human being required to make a classification, we look at their background, their education, their environment, the acquired competence – nearly all of this is used to build a body of evidence that this experience can be trusted to make a decision or classification. Like lifelong learning being recorded in a verifiable resumé, CNNs are trained on a repository of either a single domain, or pre-trained using a generic corpus of data, and while the outcomes may be similar, an anomaly in training or a tweak afterwards will yield untrustworthy results, which can be evaluated only extensive measurements.

The missing link is being able to see the training process, similar to recording inputs that a human will see from birth to deployment in the workplace. While we do not need to see everything, we would benefit from seeing the important triggers over time that form and tweak the neuron. This is the aim - to record and store the lifelog of the network, to see what parts of the training process were not necessary and to optimise before an exponentially wasteful underfitting of pathways becomes an unnecessary overhead in deployment. What is also missing is being able to baseline this model alongside this record, which would give a provenance. In the event of the next generation of AIF360 manipulating bias for any rational like Giskard from Asimov's Robots of Dawn, this would show if that manipulation was performed, and If the model can be trusted.

How we get to the next step is to implement a system of measuring the delta within a tolerance at multiple steps over the training. This can be established during CNN training in any of the available CNN frameworks, which also include predefined transfer models to identify the difference in a 'big bang' early training to additions to pre-trained systems, which will be quieter, but identify the domain impact on existing models. The visualisation will be challenging, as we will require the model to be visualised in 3D, as 2D models will hide the detail we are hoping to be exposed like a nebula in front of a galaxy. Existing visualisation techniques, and abstraction of the visualisation to multiple 2D images, similar to GradCAM, will help identify the initial benefit of the data produced by the measurement process.

III. Method

Method should outline how the task/experiment was carried out, including rationale for any decisions made. Details of any equipment and subjects used should be also included. Basically, you should include enough information, so that the reader could duplicate as much of the experimental conditions or design details as possible.

A. Reducing Model without Losing Character

The first part of the delivery required that a model, in this case a CNN model in Keras, could be reduced to an information block that each layer contains a summary of the layer that can record any changes in the weights and biases. The structure of a model is an

1) input shape
2) series of dense layers
3) output layers

and within each of the layers, there are is a series of layers that, depending on type, will contain a series of tuples that have a single of multi-dimensional series of weights, plus a set of biases. it is within these layers that we will extract the information. As there is potentially a lot of information in the layer, we attempted to reduce this by extracting two identifying elements

TABLE I
Values generated from evaluated model

Mean	The average value of all layers across all dimension
StdDev	An indication of the deviation of values, to assist in seeing how the values are weighted
Histogram	A Histogram of the distribution of values across the layer. This will assist in giving a more informed distribution. As there are a large number of potential entries in the dense layers, from binary to minute changes in float values, the Histogram will be split between the values smallest non-zero granularity and the largest value across 10 bins.
Name.	Name of Layer
Angle of Data	This is a single value representing the weighting of a Hash of the Layer Values (weights and biases)

B. Recording Changes

As we now have a method of identifying the model in a broken down format, the next stage is to record and apply a signature to the model. There are a number of methods of ensuring that the model is recorded. The key part is to take the series of hashes on each layer, combined with the model shape, and generate a signature. The signature is what will be recorded as an immutable value that can be regenerated every time the **exact** same model is used. With this identification, we can create a list showing the parent and child of the model.

To ensure this is done correctly, we have two applications - local and global.

C. Global Changes

For global changes there will be a globa repository containing the signature and summary of published models.

TABLE II
MODEL REFERENCE STORAGE

Model Identifier	This is the checksum generated by the model
Creation Date	This will be the timestamp that the model was first added to the repository
Parent	The model identification of the direct parent of the model.
Model Summary	A binary object that contains the items generated in the table above.

D. Local Changes

An additional benefit of the summarisation is to be able to view the model as it is being trained. This is a local filestore or database that contains a more regular record of the the summary changes. It is recommended that global model reporistory is only used for publishing models that require tracability, and not for regular changes.

In keeping with Git vocabulary, local changes are updated with the `model_add(model, [parent])` command, which takes a model and returns a signature and commits the information to a local repository.

The local repository stores the abstracted model information set that can be used for local comparisons. To commit the model to the public repository, the `model_add(model, [parent])` command is used, with this time the original model that already exists in the repository. The command will return an error if the parent does not exist.

E. Comparing Models

To ensure there is tracibility in between the models, if a parent model is defined, a *b-score* is generated with a degree of difference between the models. The score is based on two factors, the difference between the model structure, and the difference between the two primary parameters for each layer: *StdDev & Angle of Array*

F. Collision Management

In computer science, a collision or clash is a situation that occurs when two distinct pieces of data have the same hash value, checksum, fingerprint, or cryptographic digest.[1]

Due to the possible applications of hash functions in data management and computer security (in particular, cryptographic hash functions), collision avoidance has become a fundamental topic in computer science.

Collisions are unavoidable whenever members of a very large set (such as all possible person names, or all possible computer files) are mapped to a relatively short bit string. This is merely an instance of the pigeonhole principle.[1]

The impact of collisions depends on the application. When hash functions and fingerprints are used to identify similar data, such as homologous DNA sequences or similar audio files, the functions are designed so as to maximize the probability of collision between distinct but similar data, using techniques like locality-sensitive hashing.[2] Checksums, on the other hand, are designed to minimize the probability of collisions between similar inputs, without regard for collisions between very different inputs.[3]

IV. RESULTS AND DISCUSSION

This section includes presentations of the data and results in graphical and/or tabular form. Statistical analysis of the variation in measurements/results can also be presented. The findings should be explicitly related to the stated objectives of the project. Details and tasks that that may distract the reader unnecessarily should be presented in Appendices.

This is the section where you have a chance to discuss the results and draw conclusions from them - i.e. where you show your ability to think analytically about your findings.

V. CONCLUSIONS AND RECOMMENDATIONS

The conclusion of the practicum is there is currently no simple and efficient method of verifying the lifecycle of the development of the model. When models are released, either for production, competitions or uploaded to critical systems, they remain a blackbox. While developing a system for verifying the history of the model, it will allow us to both gain an insight into the development process, but also be able to check shared charactaristics between machine learning deep learning systems.

A. Recommendations

This section is usually left until the rest of the paper has been written. Conclusions are drawn in the context of the objectives of the project. They should be supported by data and results, and, if possible, compared with theory and data obtained by others in the literature (i.e. related published work). It is a chance to summarise what you have learnt from the project. Remember that human attention spans are extremely short, and the reader will appreciate a good summary, even if you feel that your conclusion is self-evident.

Recommendations are also extremely important, as they provide an opportunity to demonstrate the experience you have

gained. The ability to self-assess one's work with a view to suggesting ways things could have been carried out differently is a valuable asset.

Recommendations can also include suggestions to how the work could be expanded or extended. This is often placed in a separate section entitled "Future Work". The IEEEtran class file is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

REFERENCES

[1] T. Mahoney, K. R. Varshney, and M. Hind, *AI fairness: how to measure and reduce unwanted bias in machine learning*, OCLC: 1195889812. [Online]. Available: http://proquest.safaribooksonline.com/?fpi=9781492077664

[2] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps." [Online]. Available: http://arxiv.org/abs/1312.6034

[3] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes." [Online]. Available: http://arxiv.org/abs/1610.01644

[4] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets." [Online]. Available: http://arxiv.org/abs/1712.09913