2.3.2 Sufficiency of the learning process

Practical guidance – cross-domain

Authors: Rob Ashmore (Dstl), and Dr Radu Calinescu and Dr Colin Paterson (Assuring Autonomy International Programme)

The Model Learning stage of the ML lifecycle is concerned with creating a model, or algorithm, from the data presented to it. A good model will replicate the desired relationship between inputs and outputs present in the training set, and will satisfy both the defined safety requirements as well as non-functional requirements such as providing an output within a given time and using an acceptable amount of computational resources.

Stage Input and Output Artefacts

The key input artefact to this stage is the training data set produced by the Data Management stage. The key output artefacts are a machine-learnt model for verification in the next stage of the ML lifecycle and a performance deficit report used to inform remedial data management activities.

Activities

1. **Model Selection** - This activity decides the type and, where applicable, the variant and the structure of the model to be produced in the Model Learning stage. Numerous types of ML models are available [1, 2], including multiple types of classification models (which identify the category that the input belongs to), regression models (which predict a continuous-valued attribute), clustering models (which group similar items into sets), and reinforcement learning models (which provide an optimal set of actions, i.e. a policy, for solving, for instance, a navigation or planning problem).
2. **Training** - This activity optimises the performance of the ML model with respect to an objective function that reflects the requirements for the model. To this end, a subset of the training data is used to find internal model parameters (e.g. the weights of a neural network, or the coefficients of a polynomial) that minimise an error metric for the given data set. The remaining data (i.e, the validation set) are then used to assess the ability of the model to generalise. These two steps are typically iterated many times, with the training hyperparameters tuned between iterations so as to further improve the performance of the model. For each iteration data may be moved between the training and validation data sets.
3. **Hyperparameter Selection** - This activity is concerned with selecting the parameters associated with the training activity, i.e. the hyperparameters. Hyperparameters control the effectiveness of the training process, and ultimately the performance of the resulting model [3]. They are so critical to the success of the ML model that they are often deemed confidential for models used in proprietary systems [4]. There is no clear consensus on how the hyperparameters should be tuned [5]. Typical options include: initialisation with values offered by ML frameworks; manual configuration based on recommendations from literature or experience; or trial and

error [3]. Alternatively, the tuning of the hyperparameters can itself be seen as a machine learning task [6, 7].

4. **Transfer Learning** - The training of complex models may require weeks of computation on many GPUs [8]. As such, there are clear benefits in reusing ML models across multiple domains. Even when a model cannot be transferred between domains directly, one model may provide a starting point for training a second model, significantly reducing the training time. The activity concerned with reusing models in this way is termed transfer learning [9].

Desired Assurance Properties

From an assurance viewpoint, the models generated by the Model Learning stage should exhibit the key properties described below:

1. **Performant**—This property considers quantitative performance metrics applied to the model when deployed within a system. These metrics include traditional ML metrics such as classification accuracy, ROC and mean squared error, as well as metrics that consider the system and environment into which the models are deployed.
2. **Robust** -This property considers the model's ability to perform well in circumstances where the inputs encountered at run time are different to those present in the training data. Robustness may be considered with respect to environmental uncertainty, e.g. flooded roads, and system-level variability, e.g. sensor failure.
3. **Reusable** - This property considers the ability of a model, or of components of a model, to be reused in systems for which they were not originally intended. For example, a neural network trained for facial recognition in an authentication system may have features which can be reused to identify operator fatigue.
4. **Interpretable** - This property considers the extent to which the model can produce artefacts that support the analysis of its output, and thus of any decisions based on it. For example, a decision tree may support the production of a narrative explaining the decision to hand over control to a human operator.

Methods

Table 1 provides a summary of the methods that can be applied during each Model Learning activity in order to achieve the desired assurance properties (desiderata). Further details on the methods listed in Table 1 are available in [10].

Method	Associated activities†				Supported desiderata‡			
	Model Selection	Training	Hyperparam. Selection	Transfer Learning	Performant	Robust	Reusable	Interpretable
Use appropriate performance measures [11, 12]	✓	✔			★	★		
Statistical tests [13,14]	✓	✔			★			
Ensemble Learning [15]	✔	✔		✔	★	★		
Optimise hyperparameters [16, 17]		✔	✔		★	★		
Batch Normalization [18]		✔	✔		★	★		
Prefer simpler models [19, 20]	✔	✓			☆	★		☆
Augment training data		✔			★	★		
Regularization methods [21]		✔	✔			★		
Use early stopping		✔	✔			★		
Use models that intrinsically support reuse [22]	✔			✔			★	☆
Transfer Learning [23]	✔	✓		✔			★	☆
Use model zoos [21]	✔	✓		✔			★	
Post-hoc interpretability methods [24, 25, 26]		✔						★

†✔ = activity that the method is typically used in; ✓= activity that may use the method
‡★ = desideratum supported by the method; ☆ = desideratum partly supported by the method

Table 1 – Assurance methods for the Model Learning stage

Summary of Approach

1. Take the training data set produced by the Data Management stage (guidance on data management is provided in section 2.3.1)
2. Apply appropriate methods in order to undertake each activity of the model learning process to ensure a machine-learnt model is generated that achieves the desired assurance properties.
    a. Apply appropriate methods for model selection
    b. Apply appropriate methods for training
    c. Apply appropriate methods for hyperparameter selection
    d. Apply appropriate methods for transfer learning
3. Provide a machine-learnt model for verification (guidance on model verification is provided in section 2.3.3) and a performance deficit report to inform remedial data management activities.

References

- [1] Azure-Taxonomy 2019. How to choose algorithms for Azure Machine Learning Studio. Retrieved February 2019
- [2] Scikit-Taxonomy 2019. Scikit - Choosing the right estimator. Retrieved February 2019
- [3] Philipp Probst, Bernd Bischl, and Anne-Laure Boulesteix. 2018. Tunability: Importance of hyperparameters of machine learning algorithms. (2018). arXiv:1802.09596
- [4] Binghui Wang and Neil Zhenqiang Gong. 2018. Stealing hyperparameters in machine learning. In 2018 IEEE Symp. On Security and Privacy (SP). IEEE, 36–52.
- [5] Gustavo A Lujan-Moreno et al. 2018. Design of experiments and response surface methodology to tune machine learning hyperparameters, with a random forest case-study. Expert Systems with Applications 109 (2018), 195–205.

- [6] Frank Hutter, Jörg Lücke, and Lars Schmidt-Thieme. 2015. Beyond manual tuning of hyperparameters. KI-Künstliche Intelligenz 29, 4 (2015), 329–337.
- [7] Steven R Young, Derek C Rose, Thomas P Karnowski, et al. 2015. Optimizing deep learning hyper-parameters through an evolutionary algorithm. In Workshop on Machine Learning in High-Performance Computing Environments. ACM.
- [8] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. (2017). arXiv:1708.06733
- [9] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. Deep learning. Vol. 1. MIT Press.
- [10] Ashmore, R., Calinescu, R. and Paterson, C., 2019. Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges. arXiv preprint arXiv:1905.04223.
- [11] Peter Flach. 2019. Performance Evaluation in Machine Learning: The Good, The Bad, The Ugly and TheWay Forward. In 33rd AAAI Conference on Artificial Intelligence.
- [12] Kiri Wagstaff. 2012. Machine learning that matters. (2012). arXiv:1206.4656
- [13] Tom M. Mitchell. 1997. Machine Learning. McGraw-Hill.
- [14] Kevin P. Murphy. 2012. Machine Learning: A Probabilistic Perspective. The MIT Press.
- [15] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8, 4 (2018), e1249.
- [16] Frank Hutter, Jörg Lücke, and Lars Schmidt-Thieme. 2015. Beyond manual tuning of hyperparameters. KI-Künstliche Intelligenz 29, 4 (2015), 329–337.
- [17] Steven R Young, Derek C Rose, Thomas P Karnowski, et al. 2015. Optimizing deep learning hyper-parameters through an evolutionary algorithm. In Workshop on Machine Learning in High-Performance Computing Environments. ACM.
- [18] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. (2015). arXiv:1502.03167
- [19] Ajaya Adhikari,DMTax, Riccardo Satta, and Matthias Fath. 2018. Example and Feature importance-based Explanations for Black-box Machine Learning Models. (2018). arXiv:1812.09044
- [20] Stuart J Russell and Peter Norvig. 2016. Artificial intelligence: a modern approach. Pearson Education Limited.
- [21] Aurélien Géron. 2017. Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. " O'Reilly Media, Inc.".
- [22] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI). IEEE Access 6 (2018), 52138–52160.
- [23] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. Journal of Big Data 3, 1 (2016), 9.
- [24] Ajaya Adhikari,DMTax, Riccardo Satta, and Matthias Fath. 2018. Example and Feature importance-based Explanations for Black-box Machine Learning Models. (2018). arXiv:1812.09044
- [25] Samantha Krening, Brent Harrison, Karen M Feigh, et al. 2017. Learning from explanations using sentiment and advice in RL. IEEE Trans. on Cognitive and Developmental Systems 9, 1 (2017), 44–55.

- [26] Aravindh Mahendran and Andrea Vedaldi. 2015. Understanding deep image representations by inverting them. In IEEE Conf. on computer vision and pattern recognition. 5188–5196.