Towards a visual framework for the incorporation of knowledge in the phases of machine learning

CJ Swanepoel and KM Malan

Department of Decision Sciences, University of South Africa. swanecj@unisa.ac.za

Abstract. Incorporating domain knowledge into machine learning algorithms to some extent is almost unavoidable. Doing it well and explicitly can avoid unnecessary bias, improve efficiency and accuracy, and increase transparency. To increase an awareness of the relative contributions of domain knowledge and machine learning expertise, as well as an indication of the direction of information flow, a tentative qualitative visualisation framework is suggested, and two examples are given. It is hoped that such a mechanism will encourage reflection on the (sometimes implicit and innate) inclusion of domain knowledge in machine learning systems.

Keywords: Domain knowledge · Machine learning · Characterisation framework · Visualisation.

1 Introduction

Machine learning as a component of artificial intelligence, and especially deep learning, has experienced phenomenal growth over the last couple of years. (For example, in 1998 there were two papers with primary subcategory 'Machine learning' on arXiv, and in 2017 there were 2 332 papers [19].) This growth is the result of the advances in computing power, especially through the use of graphics processing units and more recently tensor processing units, the ubiquity of available data, the connectivity afforded by the internet, and major advances in machine learning algorithms by Bengio, Hinton and LeCun, amongst others [14]. In well defined domains where data is plentiful or cheap to generate and where the context is stable, machine learning can be an extremely useful tool.

However, many machine learning techniques (and especially deep learning) are fragile, greedy, shallow and not transparent [8, 12, 27]. The techniques are fragile because transfer to a slightly different domain or context usually breaks them; greedy because they require huge amounts of (labelled) data; shallow because they depend on superficial features and do not possess an underlying model based on the physical reality; and not transparent because the internal structure is often too complex to analyse and connect to the features that determine the output. This necessitates careful attention to subtle aspects of the machine learning development process.

This paper focusses on the inclusion of domain knowledge in machine learning, because this is one factor that can affect the fragility, greediness, shallowness and lack of transparency of machine learning. The aim is to provide a framework to assist practitioners to make explicit the different ways in which domain knowledge and machine learning expertise are incorporated in the stages of machine learning. Being aware of subtle inclusions of domain knowledge and innate knowledge endowed to the machine learning system might assist in identifying opportunities to improve its performance, accuracy or even transparency, and can contribute to more accurate reporting of machine learning system designs.

2 The inclusion of domain knowledge

The recent successes of machine learning that depend on data only, and use 'no domain specific' data (e.g. AlphaZero) create the impression that domain specific knowledge is not really necessary in machine learning, and might even reduce the 'generality' of the resulting system. The practical difficulties in finding useful representations of expert knowledge, and the fact that domain knowledge is often not complete or perfect, reinforces this view [27].

Two seemingly diametrically opposed views are expressed in recent literature [15, 13]: on the one hand, the view that all problems can be solved by scaling the model up and rely on the data only (AlphaZero, for example), and on the other hand, the belief that using a combination of data and domain knowledge will eventually prove to be the best approach (an idea already propounded by Alan Turing in his 1950 paper [23]). The 'data only' camp demonstrated spectacular results, particularly in the domain of game play and text generation, although the fear exists that it will not generalise easily (mainly due to data constraints in most domains) and that the performance will hit a ceiling.

Even when domain knowledge is not explicitly injected (the 'data only' approach), implicit domain knowledge almost always features in machine learning [9]. This implicit or innate domain knowledge can include the choice and structure of the algorithm, representational formats, and innate knowledge or experience [13]. For example, the choice of data encoding method depends partly on an understanding of the problem domain, and can greatly influence the effectiveness of the algorithm (see, e.g. [7, 18]). Feature selection is often task dependent and sometimes even based on intuition [5, 26]. The type of algorithm used also depends heavily on the nature of the problem domain. Marcus [13] quotes Pedro Domingos: "[Machine learning] paradigms differ in what assumptions they encode, and what form of additional knowledge they make it easy to encode."

The structure of a deep learning neural network is influenced by the nature of the problem. In the description of the neural network architecture for AlphaGo Zero, where emphasis was placed on using as little as possible explicit domain knowledge, it is stated that *"History features X_t, Y_t are necessary, because Go is not fully observable solely from the current stones. . . "* [21]. Further examples of

the extensive embedding of human domain specific knowledge in the construction of AlphaGo Zero and AlphaZero are given in [13].

In a genetic algorithm the choice of cross-over and mutation mechanisms will to some extent depend on implicit domain knowledge [10]. Risk or loss functions include implicit domain knowledge, but can also encode selected prior knowledge [15, 22]. Constraining the output of a machine learning system based on heuristics or rules from the problem domain is common practice. (For example, all Go playing algorithms before AlphaGo Zero routinely removed stupid (but legal) moves [21].)

In most (non-game) domains there are additional considerations for including domain knowledge, such as the fact that data can be difficult to get, or expensive, or might include bias, or be unbalanced (for example, the lack of edge cases). It is often necessary to explicitly include additional information to get to a feasible, unbiased or useful solution [4, 6]. The explicit inclusion of domain knowledge can potentially also improve the transparency of the model [28]. Conversely, including less explicit domain knowledge might lead to more general algorithms.

There are some obstacles to the inclusion of domain knowledge, though. These include the difficulty of finding workable encodings and injection mechanisms, the fact that most domain experts are not data science experts and *vice versa* [27]. The restrictions introduced by injecting domain knowledge can potentially also prevent the discovery of valid but unexpected solutions [3].

The explicit integration of knowledge into machine learning is called 'informed machine learning' by von Rueden *et al.* [25]. They developed a taxonomy for the explicit integration of knowledge. Their proposed taxonomy contains three components: the type of knowledge, the method used to integrate the knowledge into the machine learning system, and lastly where in the machine learning pipeline the integration happens. It is a useful tool to classify papers in the *assisted* or *informed* machine learning domain.

However, the *implicit* inclusion of domain knowledge in the form of innate knowledge and convention or experience is often not recognised and neglected in the reporting on and meta-analysis of machine learning algorithms. Many of the choices made throughout all phases of the machine learning process are based on some understanding of aspects of the problem or task, augmented by expertise and experience in the machine learning domain. This paper attempts to provide a tentative high level framework to visually characterise the inclusion of any domain knowledge into machine learning.

3 Proposed framework

In Figure 1 the generic phases of the machine learning pipeline are given as coloured blocks labelled with a capital letter. A very brief description of the phases is given in Table 1, and a more comprehensive account is given in Section 4.

Although the phases are listed more or less in the order in which they occur in the machine learning pipeline, it is an idealised representation that does not necessarily reflect the actual workflow of a specific system – in practice the order in which the phases are executed can be convoluted, and might include several iterative loops. The colours group the phases into six clusters: problem formulation (green), data preparation (yellow), machine learning activities (blue), output constraints (red), interpretation and explanation (brown), and external comparison (green).

A subjective qualitative estimate of the magnitude of the contribution from respectively domain knowledge and machine learning is represented by the relative thicknesses of the arrows in the figure. In cases where the emphasis is on the exploitation of data only, many of the arrows from domain knowledge will be thin or have zero thickness, for example. The direction of the arrows indicates the direction of information flow. Such a representation will be unique to every particular instance of a machine learning system, and can give a quick visual overview of the nature of the interactions in that instance.

The representation is subjective, and as such cannot provide accurate or quantitative data. However, this simplified model allows a quick high-level evaluation of the relative contributions from the two knowledge domains.

A: Problem identification and formulation	H: Machine learning algorithm structure determination
B: Data sourcing/labelling	I: Learning process mechanisms
C: Data cleaning/validation/quality evaluation	J: Hyperparameter tuning
D: Data augmentation	K: Constraining outputs
E: Data encoding	L: Interpretation/validation
F: Machine learning algorithm selection	M: Explanation
G: Feature engineering	N: Comparative evaluation

Table 1. Phases of machine learning

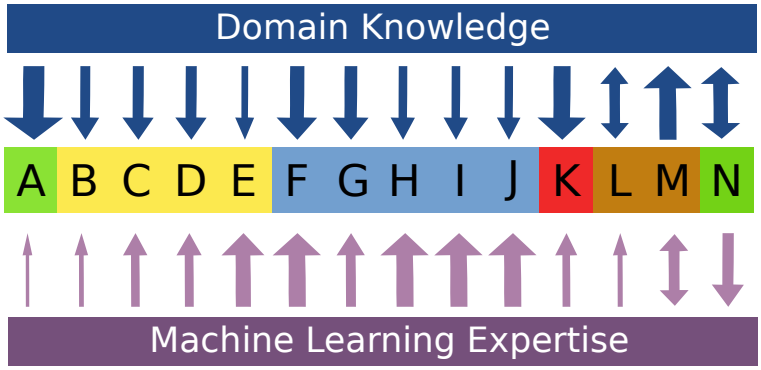

Fig. 1. The contribution of domain knowledge and machine learning expertise to a typical machine learning workflow

4 Phases of the machine learning pipeline

A: Problem identification and formulation Thorough knowledge of the domain is necessary for this step. The context will determine what data can be collected, what the objective(s) is (are), and what information might be available that is not included in the data.

B: Data sourcing/labelling Although the machine learning process often starts with available data, in some cases data will have to be sourced or labelled. Domain knowledge in the form of deep knowledge of the relationships between different features, the nature of the data, the difficulty and cost of labelling as well as an understanding of machine learning algorithms and how the data will be used can contribute to the quality of the data that is eventually used, and hence influence the outcome or success of the machine learning process.

C: Data cleaning/validation/quality evaluation Knowledge of the properties of the domain and the data collection methodology can assist in identifying outliers or invalid data points, and allow for an evaluation of the quality of the data. It will also give insight into the coverage of the data space (whether there are areas where data is too sparse to be useful).

D: Data augmentation Both knowledge of the domain and the machine learning algorithm is required for the successful augmentation of data. The imputation of missing data points, for example, can take various forms, and some of the techniques might be counterproductive in the training of a model. Another example: the perturbation of images by shifting a few pixels horizontally or vertically to provide additional training data assumes an understanding that such a translation will indeed provide novel information to the machine learning system, while it remains valid as a representation and does not influence the labelling of the image.

E: Data encoding Mainly machine learning expertise is required. However, a deep understanding of the domain knowledge environment is assumed. For example, one-hot encoding can be difficult or impossible to work with when the feature space is very large. Piech *et al.* [17] address this data encoding challenge by utilising a random low-dimensional representation of a one-hot high-dimensional vector. This encoding is motivated by the idea of *compressed sensing* introduced by Baraniuk in 2007 [1] as an effective method to capture and represent compressible signals at a rate significantly below the Nyquist rate. Another beautiful example is given in the paper by Lusci *et al.* [11] where molecules are represented as ensembles of directed acyclic graphs as input to a recursive neural network to predict the solubility of these molecules.

F: Machine learning algorithm selection This deals with the choice of a suitable machine learning algorithm. A deep understanding of the working of different machine learning techniques is required, as well as an understanding

of the nature of the problem domain. For example, if the problem has a temporal component, a recurrent neural network might be a good fit, or if filtering is required, an auto-encoder should be considered. Also see the paper by Olson *et al.* [16] where thirteen machine learning algorithms are compared over 165 publicly available classification problems.

G: Feature engineering Feature engineering includes binning, transformation of features, scaling, grouping operations and feature selection. Domain knowledge in the form of an understanding of the relationships between features and the information content of different features are required.

H: Machine learning algorithm structure determination This is sometimes described as more of an art than a science. Typically the structure of the machine learning algorithm is determined empirically through experimentation. Here experience with similar or related problems, which is a form of domain knowledge, plays a huge role. For example, Chandrasekaran *et al.* [2] in a recent paper designed a machine learning predictor for density of state and charge density of a material or molecule. For the charge density, the modelling is done with a simple fully connected neural network with one output neuron. The local density of states spectrum, on the other hand, is modelled with a recurrent neural network, where the local density of state at every energy window is represented as a single output neuron (linked via a recurrent layer to other neighbouring energy windows). Domain knowledge therefore played a huge part in determining the structure of the machine learning system.

I: Learning process mechanisms (transfer function / learning mechanism / mutation operator, etc. selection) This is probably the area where there is the biggest opportunity of innovation. Novel functions for cross-over, or innovative transfer functions can hugely influence the way in which the search space is traversed.

J: Hyperparameter tuning As with stage H, this is an area that is mostly approached empirically. The performance metric used in the hyperparameter optimisation process is influenced by domain knowledge. Setting ranges for grid or random searches to optimise hyperparameters, as well as determining of which of the hyperparameters should be included in the search cannot be analytically determined, but knowledge of the hyperparameter behaviour in related problems can provide a good starting point for an empirical search strategy.

K: Constraining outputs This is one of the most important mechanisms to include domain knowledge in the machine learning process. Techniques used include the augmentation or restriction of the loss or risk function, and filtering or transforming intermediate outputs or the final output. Stewart and Ermon [22], for example, use laws from physics to constrain the output space in training a convolutional neural network to track objects without using any labelled examples.

L: Interpretation / validation A good understanding of the underlying knowledge domain will allow an evaluation of the feasibility or quality of the outcomes. In simple classification problems this is not really an issue, but for many decision support applications this is essential. The output of a machine learning algorithm might also increase our understanding of the domain, hence the possibility of a two-way arrow in the proposed framework.

M: Explanation One of the biggest criticisms against many machine learning techniques is the lack of transparency, or the ability to explain the output of the machine learning system. Legal and moral requirements dictate that in certain environments it should be possible to justify the outcomes in the light of the inputs. Here an understanding of the domain complexities as well as the machine learning mechanism is required – although this might not be sufficient in many cases. In the proposed framework the direction of the arrow towards the domain knowledge bar indicates the flow of information that might add to our understanding of the domain. In addition, an analysis of how the machine learning process obtained the outputs could add to machine learning expertise.

N: Comparative evaluation Comparing the performance of a machine learning algorithm against previously applied approaches (benchmarking) is often necessary to evaluate the performance of a new approach. This also contributes to the knowledge base of machine learning.

5 Case studies/examples

In this section two recent contributions to the machine learning environment are described briefly, and the proposed visual representations of knowledge are given for both.

5.1 *Combination of domain knowledge and deep learning for sentiment analysis*, by Vo *et al.*

In the paper 'Combination of domain knowledge and deep learning for sentiment analysis', published in 2017, Vo *et al.* [24] found that existing approaches in the application of machine learning to sentiment analysis suffer from two major drawbacks, the first of which is that until then nobody has paid attention to the different types of sentiment terms. Different domains use different terms to express positive and negative sentiments, and some words carry a higher emotive content than others. Secondly, the loss functions used previously did not include a measure of the magnitude of sentiment misclassification, and did not distinguish between different types of misclassification.

To address these two issues, they proposed using sentiment scores (learnt by quadratic programming) to augment training data; and introduced a penalty

matrix to enhance the loss function. The enhancements were applied to a standard sentiment analysis workflow using a convolutional neural network. To evaluate the success of their approach, they compared the performance of the new system with a baseline convolution neural network sentiment analyser, as well as with a traditional support vector machine based sentiment analyser. In the comparative analysis the new approach performs better than the previous versions, showing that the inclusion of the two enhancements are useful in this domain. This knowledge and the description of the novel enhancements led to the diagram in Figure 2, where the incorporation of additional domain knowledge in the 'data augmentation' (D) and 'constraining outputs' (K) phases is emphasised.

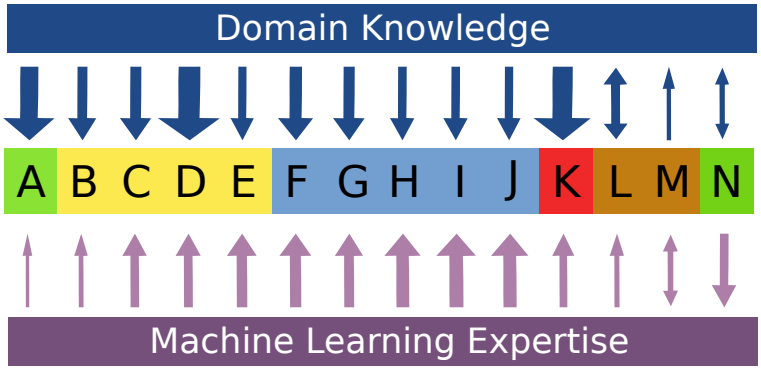

Fig. 2. The contribution of domain knowledge and machine learning expertise to the sentiment analyser of [24]

5.2 *Mastering the game of Go without human knowledge*, by Silver *et al.*

During 2016 and 2017 DeepMind released three versions of their Go playing software. The first system, now called AlphaGo Lee, defeated the world champion Lee Seedol 4–1 in a five game match in March 2016. AlphaGo Lee used two neural networks – a 'policy' and a 'value' network, as well as a Monte Carlo tree search algorithm. It was trained using historic game information, and improved through self play. A second version, AlphaGo Zero, was introduced in a paper in Nature on 19 October 2017 [21]. The title of the paper: 'Mastering the game of Go without human knowledge' expresses the major claim of this version – that it used no human or domain knowledge except for the rules of the game. A third version of the software (AlphaZero) was introduced in a paper published on arXiv in December 2017 [20]. This version added chess and shogi to the repertoire of games mastered by the system. It was claimed that in this progression the newer version each time learnt quicker and exceeded the performance of its predecessor. The contribution of knowledge into the second version, AlphaGo Zero, is discussed in this section.

In [13] Marcus extensively discusses different inclusions (some not mentioned in the original paper) of domain knowledge into the AlphaGo Zero machine learning system. He argues that the use of carefully constructed Monte Carlo tree search machinery, the artful placement of convolutional layers that allow the system to recognise that many patterns on the board are translation invariant, and the application of a sampling algorithm for dealing with reflections and rotations constitute the injection of domain knowledge into the system.

Based mostly on Marcus's assessment, rough qualitative judgements on the inclusion of domain knowledge into AlphaGo Zero were made that are reflected in Figure 3.

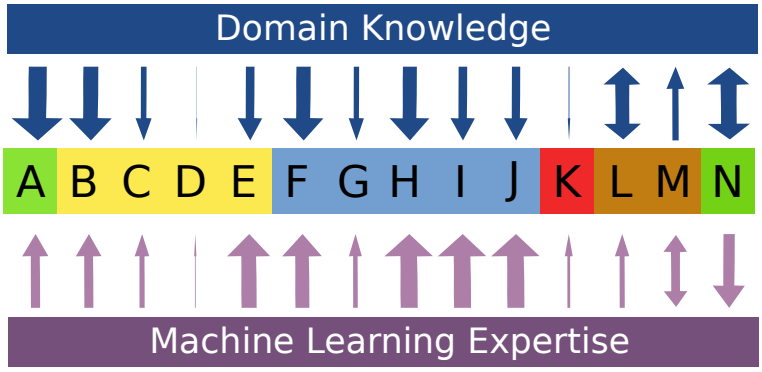

Fig. 3. The contribution of domain knowledge and machine learning expertise to AlphaGo Zero

The lack of explicit contributions from domain knowledge in the phase of 'constraining outputs' (K) is indicative of the fact that this popular method of injecting domain knowledge was not used. There are, however, substantial contributions in phases F and H, the selection of the machine learning algorithm (a combination of Monte Carlo search trees and reinforcement learning) and the structure determination of the algorithm (taking into account symmetries, for example). The direction of information flow in the three phases 'interpretation/validation' (L), 'explanation' (M) and 'comparative evaluation' (N) should also be noted. Domain knowledge and machine learning expertise are expanded to varying degrees with a successful implementation of a machine learning system. In the case of AlphaGo Zero new ('alien') gameplay strategies were discovered (domain knowledge), and some insight gained into the comparative performance of different approaches (machine learning expertise).

6 Conclusion

A visualisation scheme for the inclusion of domain knowledge and machine learning expertise into machine learning systems is proposed. The hope is that it will aid in greater awareness of the innate, implicit and explicit use of domain knowledge in the machine learning workflow.

References

1. Baraniuk, R.: Compressive sensing [lecture notes]. IEEE Signal Processing Magazine **24**(4), 118–121 (Jul 2007). https://doi.org/10.1109/msp.2007.4286571
2. Chandrasekaran, A., Kamal, D., Batra, R., Kim, C., Chen, L., Ramprasad, R.: Solving the electronic structure problem with machine learning. npj Computational Materials **5**(1) (Feb 2019). https://doi.org/10.1038/s41524-019-0162-7
3. Childs, C.M., Washburn, N.R.: Embedding domain knowledge for machine learning of complex material systems. MRS Communications pp. 1–15 (Jul 2019). https://doi.org/10.1557/mrc.2019.90
4. Choo, J., Liu, S.: Visual analytics for explainable deep learning. IEEE Computer Graphics and Applications **38**(4), 84–92 (Jul 2018). https://doi.org/10.1109/mcg.2018.042731661
5. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011), http://dl.acm.org/citation.cfm?id=2078186
6. Du, M., Liu, N., Hu, X.: Techniques for interpretable machine learning. arXiv:1808.00033 (2018)
7. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Computation **4**(1), 1–58 (Jan 1992). https://doi.org/10.1162/neco.1992.4.1.1
8. Ghorbani, A., Abid, A., Zou, J.: Interpretation of Neural Networks is Fragile. arXiv:1710.10547 arXiv:1710.10547 (Oct 2017)
9. Hessel, M., van Hasselt, H., Modayil, J., Silver, D.: On inductive biases in deep reinforcement learning. arXiv:1907.02908 (2019)
10. Johns, M.B., Mahmoud, H.A., Walker, D.J., Ross, N.D.F., Keedwell, E.C., Savic, D.A.: Augmented evolutionary intelligence. In: Proceedings of the Genetic and Evolutionary Computation Conference on – GECCO '19. ACM Press (2019). https://doi.org/10.1145/3321707.3321814
11. Lusci, A., Pollastri, G., Baldi, P.: Deep architectures and deep learning in chemoinformatics: The prediction of aqueous solubility for drug-like molecules. Journal of Chemical Information and Modeling **53**(7), 1563–1575 (Jul 2013). https://doi.org/10.1021/ci400187y
12. Marcus, G.: Deep learning: A critical appraisal. arXiv:1801.00631 (2018)
13. Marcus, G.: Innateness, AlphaZero, and artificial intelligence. arXiv:1801.05667 (2018)
14. Marx, V.: Machine learning, practically speaking. Nature Methods **16**(6), 463–467 (May 2019). https://doi.org/10.1038/s41592-019-0432-9
15. Muralidhar, N., Islam, M.R., Marwah, M., Karpatne, A., Ramakrishnan, N.: Incorporating prior domain knowledge into deep neural networks. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 36–45 (Dec 2018). https://doi.org/10.1109/BigData.2018.8621955

16. Olson, R.S., La Cava, W., Mustahsan, Z., Varik, A., Moore, J.H.: Data-driven Advice for Applying Machine Learning to Bioinformatics Problems. arXiv:1708.05070 arXiv:1708.05070 (Aug 2017)
17. Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J., Sohl-Dickstein, J.: Deep knowledge tracing. arXiv:1506.05908 (2015)
18. Potdar, K., Pardawala, T.S., Pai, C.D.: A comparative study of categorical variable encoding techniques for neural network classifiers. International Journal of Computer Applications **175**(4), 7–9 (Oct 2017). https://doi.org/10.5120/ijca2017915495
19. Shoham, Y., Perrault, R., Brynjolfsson, E., Clark, J., Manyika, J., Niebles, J.C., Lyons, T., Etchemendy, J., Grosz, B., Bauer, Z.: The AI Index 2018 Annual Report. AI Index Steering Committee, Human-Centered AI Initiative, Stanford University, Stanford, CA (Dec 2018)
20. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T.P., Simonyan, K., Hassabis, D.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv:1712.01815 (2017)
21. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., Hassabis, D.: Mastering the game of Go without human knowledge. Nature **550**(7676), 354–359 (Oct 2017). https://doi.org/10.1038/nature24270
22. Stewart, R., Ermon, S.: Label-free supervision of neural networks with physics and domain knowledge. arXiv:1609.05566 (2016)
23. Turing, A.M.: I.—Computing Machinery and Intelligence. Mind **LIX**(236), 433–460 (Oct 1950). https://doi.org/10.1093/mind/lix.236.433
24. Vo, K., Pham, D., Nguyen, M., Mai, T., Quan, T.: Combination of domain knowledge and deep learning for sentiment analysis. In: Lecture Notes in Computer Science, pp. 162–173. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-69456-6_14
25. von Rueden, L., Mayer, S., Garcke, J., Bauckhage, C., Schuecker, J.: Informed Machine Learning – Towards a Taxonomy of Explicit Integration of Knowledge into Machine Learning. arXiv:1903.12394 (Mar 2019)
26. Yang, L., Zheng, Z., Sun, J., Wang, D., Li, X.: A domain-assisted data driven model for thermal comfort prediction in buildings. In: Proceedings of the Ninth International Conference on Future Energy Systems. pp. 271–276. e-Energy '18, ACM, New York, NY, USA (2018). https://doi.org/10.1145/3208903.3208914
27. Yu, T., Jan, T., Simoff, S., Debenham, J.: Incorporating prior domain knowledge into inductive machine learning. Technical report, International Institute of Forecasters (IIF), University of Massachusetts, Amherst, USA (Oct 2006)
28. Yu, T., Simoff, S., Jan, T.: VQSVM: A case study for incorporating prior domain knowledge into inductive machine learning. Neurocomputing **73**(13-15), 2614–2623 (Aug 2010). https://doi.org/10.1016/j.neucom.2010.05.007