Incorporating Domain Knowledge into Deep Neural Networks

Tirtharaj Dash[1,3], **Sharad Chitlangia**[1,2], **Aditya Ahuja**[1,3], **Ashwin Srinivasan**[1,3]
[1]APP Centre for Artificial Intelligence Research (APPCAIR)
[2]Department of Electrical and Electronics Engineering
[3]Department of Computer Science and Information Systems
BITS Pilani, K.K. Birla Goa Campus, Goa 403726, India
{tirtharaj,f20170472,f20170080,ashwin}@goa.bits-pilani.ac.in

Abstract

We present a survey of ways in which domain-knowledge has been included when constructing models with neural networks. The inclusion of domain-knowledge is of special interest not just to constructing scientific assistants, but also, many other areas that involve understanding data using human-machine collaboration. In many such instances, machine-based model construction may benefit significantly from being provided with human-knowledge of the domain encoded in a sufficiently precise form. This paper examines two broad approaches to encode such knowledge–as logical and numerical constraints–and describes techniques and results obtained in several sub-categories under each of these approaches.

1 Introduction

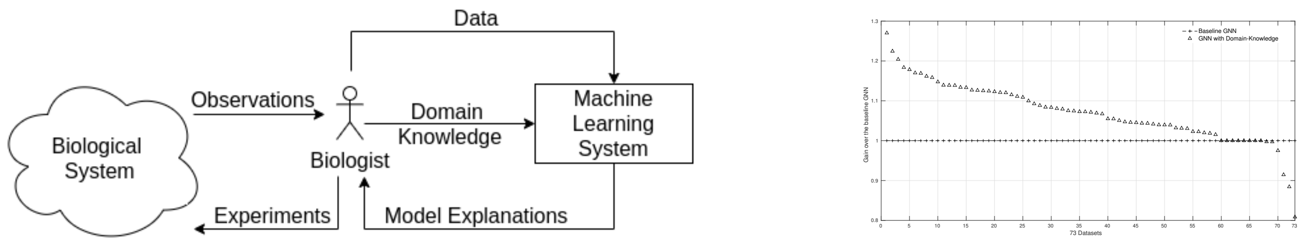
Artificial Intelligence for Science [Stevens *et al.*, 2020] is concerned with the use of AI methods to accelerate our understanding of the natural world, and to assist the application of this understanding to the development of areas of engineering, medicine, healthcare, agriculture, environment and so on. While ambitious plans exist for completely automated AI-based robot scientists [Kitano, 2016], the main use of AI for Science remains semi-automated, with a scientist-in-the-loop. An example of such a collaborative system is in Fig. 1(a). For such systems to work effectively, we need at least the following: (1) We have to be able to tell the machine what we know, in a suitably precise form; and (2) The machine has to be able to tell us what it is has found, in a suitably understandable form. While the remarkable recent successes of deep neural networks on a wide variety of tasks makes a substantial case for their use in model construction, it is not immediately obvious how either (1) or (2) should be done with deep neural networks. In this paper, we examine ways of achieving (1). Understanding models constructed by deep neural networks is an area of intense research activity, and good summaries exist elsewhere [Lipton, 2016; Arrieta *et al.*, 2019]. To motivate the utility of providing domain-knowledge to a deep network, we reproduce a result from [Dash *et al.*, 2020] in Fig. 1(b), which shows that predictive performance can increase significantly, even with a simplified encoding of domain-knowledge. It is unsurprising that a recent report on AI for Science [Stevens *et al.*, 2020] identifies the incorporation of domain-knowledge as one of the 3 Grand Challenges in developing AI systems:

> "Off-the-shelf [ML and AI] practice treats [each of these] datasets in the same way and ignores domain knowledge that extends far beyond the raw data... Improving our ability to systematically incorporate diverse forms of domain knowledge can impact every aspect of AI."

In this survey, we restrict the studies on incorporation of domain knowledge into neural networks, with 1 or more hidden layers (we will sometimes also use the term deep neural network, or DNN). Before we proceed further, we clarify that our focus here are more specific than that of research that looks at the development of hybrid systems combining neural and logical systems (see for example, [Garcez *et al.*, 2012]); and different to the use of neural network techniques for either emulating logical inference or to represent logical concepts. We refer the reader to [Besold *et al.*, 2017] for reviews of some of these other strands of work. These reviews are nevertheless relevant to some of the material in this paper since they identify some key challenges in integrating neural-based learning with symbolic knowledge representation and logical reasoning. More directly related to this paper is the work on "informed machine learning", reviewed in [von Rueden *et al.*, 2019]. We share with this work the interest in prior knowledge as an important source of information that can augment existing data. However, the goals of that paper are more ambitious than here. It aims identify categories of prior knowledge, using as dimensions: the source of the knowledge, its representation, and its point of use in a machine-learning algorithm. In this paper, we are only concerned with some of these categories. Specifically, in terms of the categories in [von Rueden *et al.*, 2019], we are interested in implicit or explicit sources of domain-knowledge, represented either as logical or numeric constraints, and used at the model-construction stage by DNNs.

Our contributions:

In this survey, we consider the representation of domain knowledge for DNNs in two broad categories: (1) as logical



(a) (b)

Figure 1: An example of AI for Science. In (a), the human-in-the-loop is a biologist. The plot in (b) from [Dash *et al.*, 2020], which shows gains in predictive accuracy of graph neural network (GNN) with the inclusion of domain-knowledge, using a simplified technique called vertex-enrichment. The results shown are over 70 datasets.

constraints; and (2) as numerical constraints.[1] Under each of these, we consider sub-categories: (1a) Propositional logic, including canonical normal forms; (1b) Predicate logic, including binary relations, and more generally, n-ary relations; (2a) Loss functions, including regularisation terms; and (2b) Weight initialisations, including priors and transfer-learning.

The rest of the paper is organised as follows: Section 2 describes inclusion of domain-knowledge as logical constraints. Section 3 describes the inclusion of domain-knowledge as numeric constraints. Section 4 outlines some major challenges related to the inclusion of domain-knowledge in DNNs with our perspectives on on the relevance of the use of domain-knowledge to aspects of Responsible AI, including ethics, fairness, and explainability of DNNs.

2 Domain-Knowledge as Logical Constraints

2.1 Propositional Logic

Although not all DNNs require data to be a set of feature-vectors, this form of data representation is long-standing and still sufficiently prevalent. In logical terms, we categorise feature-based representations as being encodings in a propositional logic. The reader would point out, correctly, that feature-values may not be Boolean. This is correct, but we can represent non-Boolean features by Boolean-valued propositions (for example, a real-valued feature f with value 4.2 would be represented by a corresponding Boolean feature f' that has the value 1 if $f = 4.2$ and 0 otherwise). With the caveat of this rephrasing, it has of course been possible to provide domain-knowledge to neural networks by employing domain-specific features devised by an expert. However, we focus here on ways in which domain-knowledge encoded as rules in propositional logic has been used to constrain the structure or parameters of models constructed by a network.

Here, domain-knowledge is encoded as a set of propositional rules. These rules in turn constrain the structure of the neural network. Weight-learning then proceeds as normal, using the structure. The result could be thought of as learning weighted forms of the antecedents present in the rules. The most popular and oldest work along this line is Knowledge-Based Artificial Neural Network (KBANN) [Towell *et al.*, 1990] that incorporates knowledge into neural networks. In KBANN, the domain knowledge is represented as a set of hierarchically structured propositional rules that directly determines a fixed topological structure of a neural network [Towell and Shavlik, 1994]. KBANN was successful in many real-world applications; but, its representational power was bounded by pre-existing set of rules which restricted it to refine these existing rules rather than discovering new rules. A similar study is KBCNN [Fu, 1993], which first identifies and links domain attributes and concepts consistent with initial domain knowledge. Further, KBCNN introduces additional hidden units into the network and most importantly, it allowed decoding of the learned rules from the network in symbolic form. However, both KBANN and KBCNN were not appropriate for learning new rules because of the way the initial structure was constructed using the initial domain knowledge base.

Some of the limitations described above could be overcome with the proposal of a hybrid system by Fletcher and Obradovic [Fletcher and Obradovic, 1993]. The system was able to learn a neural network structure that could construct new rules from an initial set of rules. Here, the domain knowledge is transformed into an initial network through an extended version of KBANN's symbolic knowledge encoding. It performed incremental hidden unit generation thereby allowing construction or extension of initial rule-base. In a similar manner, there was a proposal for using Cascade ARTMAP [Tan, 1997] which could not only construct a neural network structure from rules but also perform explicit cascading of rules and multistep inferencing. It was found that the rules extracted from Cascade ARTMAP are more accurate and much cleaner than the rules extracted from KBANN [Towell and Shavlik, 1993].

In the late 1990s, Garcez and Zaverucha proposed a massively parallel computational model called CIL^{2}P based on feedforward neural network that integrates inductive learning from examples and domain knowledge, expressed as a propositional logic program [Avila Garcez and Zaverucha, 1999]. A translation algorithm generates a neural network. Unlike KBANN, the approach uses the notion of "bipolar semi-linear" neurons. This allows the proof of a form of cor-

[1]We use the term "constraints" here in the sense that the domain-knowledge constrains either the structure or parameters (or both) of a DNN.

rectness, showing the existence of a neural-network structure that can compute the logical consequences of the domain-knowledge. The output of such a network, when combined into subsequent processing naturally incorporates the intended interpretation of the domain predicates. The authors extend this to the use of first-order logic programs: we consider this in a later section.

A recent proposal focuses on embedding symbolic knowledge expressed as logical rules [Xie *et al.*, 2019]. It considers two languages of representations: Conjuctive Normal Form (CNF) and decision-Deterministic Decomposable Negation Normal form (d-DNNF), which can naturally be represented as graph structures. The graph structures can be provided to a graph neural network (GNN) to learn an embedding suitable for further task-specific implementations.

Somewhat in a similar vein to the work by [Avila Garcez and Zaverucha, 1999], the work reported in [Xu *et al.*, 2018] considers as a set of propositional statements representing domain constraints. A deep network is then trained to find satisfying assignments for the constraints. Again, once such a network is constructed, it can clearly be used in subsequent processing, capturing the effect of the domain constraints. The network is trained using a semantic loss that we describe in a later section.

2.2 First-Order Logic

We now describe methods that use domain-knowledge described in first-order logic, which clearly gives greater flexibility and representational power than the use of propositional logic. Although there is no conceptual need to do so, we nevertheless first consider the encoding of domain-knowledge as binary relations, and later consider the general case. The reasons for this will be apparent below.

Binary Relations

An influential form of representing domain-knowledge takes the form *knowledge graph*, which are labelled graphs, with vertices representing entities and edges representing relations between entities. In essence, this represents a binary relation. We refer the reader to [Hogan *et al.*, 2021] to a comprehensive survey of this form of representation for domain-knowledge.

Incorporation of the information in a knowledge-graph into deep neural models–termed "knowledge-infused learning"–is described in [Kursuncu *et al.*, 2019; Sheth *et al.*, 2019]. This aims to incorporate binary relations contained in application-independent sources (like DBPedia, Yago, WikiData) and application-specific sources (like SNOMED-CT, DataMed). The work examines techniques for incorporating relations at various layers of deep-networks (the authors categorise these as "shallow", "semi-deep" and "deep" infusion). In the case of shallow infusion, both the external knowledge and the method of knowledge infusion is shallow, utilising syntactic and lexical knowledge in the form of word embedding models. In semi-deep infusion, external knowledge is involved through attention mechanisms or learnable knowledge constraints acting as a sentinel to guide model learning, and deep infusion employs a stratified representation of knowledge representing different levels of abstractions in different layers of a deep learning model, to transfer knowledge that aligns with the corresponding layer in the layered learning process.

Knowledge graphs can be encoded directly for use by a graph neural network (GNN). The computational machinery available in GNN then aggregates and combines the information available in the knowledge graph. The final collected information from this computation could be used for further predictions. Some recent works are in [Park *et al.*, 2019; Wang *et al.*, 2019], where a GNN is used for estimation of node importance in the knowledge-graph. The idea of encoding a knowledge graph directly for a GNN is also used in [Chen *et al.*, 2019] to enrich the information provided in the knowledge-graph.

n-ary Relations

The pre-eminent form of symbolic machine learning based on the use of relations in first-order logic is Inductive Logic Programming (ILP) [Muggleton, 1991], which has an explicit role for domain-knowledge being incoprated into learning. In ILP, domain-knowledge is represented in first-order logic: we refer the reader to surveys, both old and new, for a description of the field [Muggleton and de Raedt, 1994; Muggleton *et al.*, 2012; Cropper *et al.*, 2020].

The simplest use of ILP to incorporate n-ary relations in domain knowledge into a neural network relies on techniques that automatically "flatten" the domain-knowledge into a domain-specific propositional representation. Techniques for automatic construction of Boolean-valued features from relational domain-knowledge have a long history in the field of ILP, often called *propositionalisation*, originating from the LINUS [Lavrač *et al.*, 1991]. This involves the construction of features that identify the conditions under which they take on the value 1 (or 0). For example, given (amongst other things) the definition of benzene rings and of fused rings, an ILP-based propositionalisation may construct the Boolean-valued feature that has the value 1 if a molecule has 3 fused benzene rings, and 0 otherwise. The values of such Boolean-valued features allows us to represent a data instance (like a molecule) as a Boolean-valued feature-vector, which can then provided to a neural network. There is a long history of propositionalisation: see [Kramer *et al.*, 2001] for a review of some of early use of this technique, and [Lavrac *et al.*, 2020; Vig *et al.*, 2017] who examine the links between propositionalisation and modern-day use of embeddings in neural-networks.

A direct application of propositionalisation, demonstrating its utility for deep networks has been its use in Deep Relational Machines [Lodhi, 2013]. A DRM is a deep fully-connected neural network with Boolean-valued inputs obtained from propositionalisation by an ILP engine. In [Dash *et al.*, 2018] Boolean-valued features from an ILP engine are sampled from a large space of possible features. The sampling technique is refined further in [Dash *et al.*, 2019].

The idea of propositionalisation also forms the foundation for a method known as 'Bottom Clause Propositionalisation (BCP)' to propositionalise literals of a most-specific clause, or "bottom-clause". Given a data instance, the bottom-clause is the most-specific first-order clause that entails the data instance, given some domain-knowledge. Loosely speaking, the most-specific clause can be thought of "enriching" the data instance with all domain relations that are true, given the data instance. The construction of such most-specific clauses and their subsequent use in ILP was introduced in [Muggleton, 1995]. CILP++ [França *et al.*, 2014] uses bottom-clauses for data instances to construct feature-vectors for neural networks. This is an extension to CIL^{2}P. Here the neural network has recurrent connections.

Propositionalisation has conceptual and practical limitations. Conceptually, there is no variable sharing between two or more first-order features [Dash *et al.*, 2018]. Practically, the space of possible features can be extremely large: this has meant that the feature-selection has usually been done separately from the construction of the neural network. There are some recent reports on incorporating first-order logic that do not rely on propositionalisation. In [Li and Srikumar, 2020] it is proposed to augment a language model that uses a deep net architecture with additional statements in first-order logic. Thus, given domain-knowledge encoded as first-order relations, connections are introduced into the network, based on the logical constraints enforced by the domain-relations. The approach is related somewhat to the work in [Sourek *et al.*, 2018] that does not explicitly consider the incorporation of domain-knowledge but does constrain a deep neural network's structure based on the relational structure within data instances.

A work that does not employ either propositionalisation or network augmentation considers a combination of symbolic knowledge represented in first-order logic with matrix factorization techniques [Rocktäschel *et al.*, 2015]. This exploits dependencies between textual patterns to generalise to new relations. In another study, drawing on representation of n-ary relations as hyperedges, the technique of *vertex enrichment* is proposed in [Dash *et al.*, 2020]. This provides a simplified way to incorporate symbolic domain-knowledge into standard graph neural networks (GNNs) [Dash *et al.*, 2020]. Vertex enrichment incorporates first-order background relations as additional features into the features associates with the nodes of a graph provided to a GNN. The results reported in the paper show significant improvements in the predictive accuracy of GNNs across a large number datasets.

We note that newer areas are emerging that use representations for domain-knowledge that go beyond first-order logic relations. This includes probabilistic first-order logic, as a way of including uncertain domain-knowledge [Manhaeve *et al.*, 2018]. One interesting way this is being used is to constrain the training of "neural predicates", which represent probabilistic relations that are implemented by neural networks.

3 Domain-Knowledge as Numerical Constraints

We now discuss some forms of numeric constraints often used as ways of incorporating domain-knowledge into a neural network. Some of the methods are standalone methods and some of these methods are coupled with methods incorporating logical constraints to provide a more robust way of integration of domain-knowledge into a network.

3.1 Utility or Loss Functions

A fairly standard way of incorporating domain-knowledge into a deep network is by introducing additional loss terms into the utility (loss) function that the network optimises. This function takes on a general form like the following:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{task} + \beta\mathcal{L}_{in} + \gamma\mathcal{L}_{out} + \lambda\mathcal{L}_{model},$$

where $\mathcal{L}_{total}$ is the total loss for the deep network-based on which training will be carried out, $\mathcal{L}_{task}$ denotes to the standard task-specific loss (e.g. cross-entropy for classification, mean-squared-error for regression, and so on), $\mathcal{L}_{in}$ is the loss representing some constraints on the inputs, and $\mathcal{L}_{out}$ is the loss representing constraints on outputs, $\mathcal{L}_{model}$ is the constraint on the function that is to be learned by the deep network. The parameters $\alpha, \beta, \gamma, \lambda$ are the constants signifying some form of weights for each of the above loss terms, which can be learned or tuned during model construction.

A recent work that is based on loss function is in [Xu *et al.*, 2018]. Here the authors propose a semantic loss that signifies how well the outputs of the deep network matches some given constraints encoded as propositional rules. The general intuition behind this idea is that the semantic loss is proportional to a negative logarithm of the probability of generating a state that satisfies the constraint when sampling values according to some probability distribution. This loss function falls under the category of $\mathcal{L}_{out}$. This kind of loss function is particularly useful for semi-supervised learning as these losses behave like self-information and are not constructed using explicit labels and can thus utilize unlabelled data.

[Hu *et al.*, 2016] proposed a framework to incorporate first-order logic rules with the help of an iterative distillation procedure that transfers the structured information of logic rules into the weights of neural networks. This is done via a modification to the knowledge-distillation loss proposed by Hinton et al. [Hinton *et al.*, 2015]. The authors show that taking this loss-based route of integrating rule-based domain-knowledge allows the flexibility of choosing a deep network architecture suitable for the intended task.

In [Fischer *et al.*, 2019], authors construct a system for training a neural network with domain-knowledge encoded as logical constraints. Here the available constraints are transferred to a loss function. Specifically, each individual logic operation (such as negation, and, or, equality etc.) is translated to a loss term. The final formulation results in an optimisation problem. The authors extract constraints on inputs that capture certain kinds of convex sets and use them as optimisation constraints to make the optimisation tractable. In the developed system, it is also possible to pose queries on the model to find inputs that satisfy a set of constraints.

3.2 Constraints on Weights

In Bayesian framework, explicitly information about a machine learning model and data can be expressed succinctly in the form:

$$\text{posterior} \propto \text{prior} \cdot \text{sample-likelihood}$$

A very common way of incorporating domain-knowledge into a machine learning system (including deep networks) is by encoding it as a 'prior' term in the above Bayes equation. The domain-knowledge here could be about the problem, the neural network structure, or the neural network parameters, that is, some form of a probability distribution over these. The priors on networks and network weights represent our expectations about networks before receiving any data, and correspond to penalty terms or regularisers. Buntine and Weigend [Buntine and Weigend, 1991] extensively study how Bayesian theory can be highly relevant to the problem of training feedforward neural networks. One of the main focus of this study was to study principles of choosing an appropriate network structure and size based on prior domain-knowledge about the problem and also of selecting a prior on the weight parameters.

Priors

Seminal work by [Neal, 1995] on Bayesian learning in neural networks showed how domain-knowledge could help build a prior probability distribution over neural network parameters. They showed how with the approach of Bayesian learning, networks are self-regularised to not over-fit even when complexity of the neural network is increased to infinity. This study has served as foundation to various works on regularisation approaches such as posterior regularisation in the case of neural networks. In a similar spirit, [Krupka and Tishby, 2007] showed how prior domain knowledge could be used to define 'meta-features' that can aid in defining the prior distribution of weights. These meta-features are additional information about each of the features in the available data. For instance, for an image recognition task, the meta-feature could be the relative position of a pixel (x, y) in the image. This meta information can be used to construct a prior over the weights for the original features.

3.3 Transfer Learning

Transfer Learning is a very common technique utilised when there is fewer data in the target domain pertaining to the prediction task; and lots of data in the domain of a task similar to the target domain. From the Bayesian perspective, transfer learning allows the construction of the prior over the weights of a neural network for the target domain based on the posterior constructed in the source domain. Transfer learning is not limited by the kind of task (such as classification, regression, etc) but rather by the domain itself. This is a result of the multi-layered information learnt by neural networks i.e., the final layer, which is most often corresponding to the task in hand, can be replaced and fine-tuned separately [Yosinski *et al.*, 2014]. Large language models are a very successful example of this, where the models are initially learnt on a huge corpus of data and fine-tuned for numerous tasks. [Zhuang *et al.*, 2020] provides an in-depth review of some of the mechanisms and the strategies of transfer learning. Transfer learning need not be restricted to deep networks only: in a recent study, [Liu *et al.*, 2018] proposes a model that transfers knowledge from a neural network to a decision tree using knowledge distillation framework. The symbolic knowledge encoded in the decision tree could further be utilised for further for a variety of tasks.

3.4 Regularisation

A natural consequence of incorporating prior knowledge (some knowledge on the prior distribution) results in regularisation in learning. This allows the inductive bias to be exploited based on prior knowledge and further allowing better predictive performance. One of the most common regularisations is a penality based regularisation for model complexity. Examples include L_1 or L_2 based regularisation terms in the utility or loss function of a neural network. The optimiser minimises the loss along with the regularisation term resulting in a less complex model in terms of parameters [Kukačka *et al.*, 2017]. [Li *et al.*, 2018] show that domain-based regularisation in loss function can also help in constructing deep networks with less amount of data.

Over the years, regularising embedding constitutes another major direction of research. [Fu, 1995] was one of the earliest works in this domain, in which they proposed a strategy to establish constraints by designating each node in a Hopfield Net to represent a concept and edges to represent their relationships and learn these nets by finding the solution which maximises the greatest number of these constraints. [Rocktäschel *et al.*, 2014] was perhaps the first method of regularising embeddings from declarative knowledge encoded in first-order logic. [Rocktäschel *et al.*, 2015] extended this to regularisation by addition of differentiable loss terms to the objective-based on propositionalisation of each first-order predicate. [Li and Srikumar, 2020] develop a method to constraint individual neural layers using soft logic based on massively available declarative rules in ConceptNet. [Hamilton *et al.*, 2018] incorporate first-order logic into low dimensional spaces by embedding graphs nodes and represent logical operators as learned geometric relations in the space. [Demeester *et al.*, 2016] proposed ordering of embedding space based on rules mined from WordNet and find it to better prior knowledge and generalisation capabilities using these relational embeddings. [Silvestri *et al.*, 2020] probe embeddings of a neural network while progressively adding domain knowledge and show strong results encouraging the same. In [Takeishi and Akimoto, 2018], a knowledge-based distant regularisation framework was proposed, in which distance domain information encoded in a knowledge graph was utilised. It defines prior distributions of model parameters using knowledge graph embeddings. They show that this results in an optimisation problem for a regularised factor analysis method.

4 Challenges and Concluding Remarks

We summarise our discussion on domain-knowledge as constraints in Table 1. We now outline some challenges in incorporating domain-knowledge encoded as logical or numerical constraints into a deep network. We first outline some immediate practical challenges concerning the logical constraints:

- There is no standard framework for translating logical constraints to neural networks. While there are simplification methods which first construct a representation of the logical constraint that a standard deep network can consume, this process has its limitations as described in the relevant section above.

Work	Logical Constraint	Numerical Constraint	DNN Method
KBANN [Towell and Shavlik, 1994]	Propositional Logic	-	MLP
Cascade-ARTMAP [Tan, 1997]	Propositional Logic	-	ARTMAP
CIL^{2}P [Avila Garcez and Zaverucha, 1999]	Propositional Logic	-	RNN
LENSR [Xie *et al.*, 2019]	Canonical Normal Form	Loss function	GNN
CILP++ [França *et al.*, 2014]	First-order Logic	-	MLP
DRM [Lodhi, 2013]	First-order Logic	-	MLP
VEGNN [Dash *et al.*, 2020]	First-order Logic	-	GNN
Semantic Loss [Xu *et al.*, 2018]	First-order Logic	Loss function	CNN
HDNNLR [Hu *et al.*, 2016]	First-order Logic	Knowledge Distillation	CNN, RNN
DL2 [Fischer *et al.*, 2019]	-	Loss function	CNN,
ILBKRME [Rocktäschel *et al.*, 2015]	-	Loss function, Regularisation	MLP
IPKFL [Krupka and Tishby, 2007]	-	Prior	CNN

Table 1: Summarising some selected works based on the kinds of domain knowledge used, and deep network method.

- Logic is not differentiable. This does not allow using standard training of deep network using gradient based methods in an end-to-end fashion. Propagating gradients via logic has now been looked at in [Evans and Grefenstette, 2018], but the solution is intractable and does not allow day-to-day use.
- Neural networks are directed acyclic graphs (DAGs). However, logical formula can introduce cyclic dependencies, which needs a separate form of translations.

There are also practical challenges concerning the numerical constraints:

- We have seen that the numerical constraints are often provided with the help of modification to a loss function. Constructing a term in loss function is not straightforward.
- The process of introducing a loss term often results in a difficult optimisation problem (sometimes constrained) to be solved. This may require additional mathematical tools for a solution that can be implemented practically.

The Domain-Knowledge Grand Challenge

Incorporating domain-knowledge into learning is highlighted in [Stevens *et al.*, 2020] as one of the Grand Challenges facing the foundations of AI and ML. The principal difficulties raised in that report are these:

- "Can the constructed deep network model be trusted?" This question involves long-standing discussions on explainability and interpretability of deep models. It also includes the question of whether data used for constructing the deep model contains sufficient information without introducing spurious correlations or bias that would invalidate the model itself.
- "Why does the AI model work for a problem?" To address this question, there has to be some a mapping between the internal representation of the model to a domain-specific concept. In [Futia and Vetrò, 2020], authors identify that the knowledge mapping of the deep learning components, including input features, hidden unit and layers, and output predictions with domain-knowledge could lead to an understandable model.

Going Beyond Prediction

The issues raised above go beyond just the "how" questions related to the incorporation of domain-knowledge into deep networks. They provide pointers to why the use of domain-knowledge may extend beyond its utility for prediction. Domain-knowledge can also play a role in aspects like explanation and fairness. We mention some of the challenges that result.

One important requirement of machine-constructed models in workflows with humans-in-the-loop is that the models are human-understandable. Domain-knowledge can be used in two different ways to assist this. First, it can constrain the kinds of models that are deemed understandable. Secondly, they can provide concepts that are meaningful for use in a model. Most of the work in this review has been focussed on improving predictive performance. However, the role of domain-knowledge in constructing explanations for deep network models is also being explored (see for example, [Srinivasan *et al.*, 2019]). However, that work only generates *post hoc* explanations that are locally consistent. Explanatory deep network models that identify true causal connections based on concepts provided as domain-knowledge remain elusive.

Domain-knowledge can also be used to correct biases built into a deep network either declaratively, through the use of constraints, or through the use of loss functions that include"ethical penalty" terms. Demonstrations of the use of domain-knowledge driven, ethics-sensitive machine learning have been available in the literature for some time [Anderson *et al.*, 2005]. Can these carry over to the construction of deep network models? This remains to be investigated.

Finally, the rapid progress in the area of language models raises the possibility of providing domain-knowledge in forms other than logical or numerical. While the precision of these formal representations may continue to be needed for the construction of scientific assistants, their role in representing commonsense knowledge is less evident. Day-to-day machine assistants that can incorporate informal knowledge of the world will be needed. Progress in this is being made (see for example, https://allenai.org/aristo), but there is much more that needs to be done to make the language models required accessible to everyday machinery.

References

[Anderson *et al.*, 2005] Michael Anderson, S. Anderson, and Chris Armen. Medethex: Toward a medical ethics advisor. In *AAAI Fall Symposium: Caring Machines*, 2005.

[Arrieta *et al.*, 2019] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *arXiv preprint arXiv:1910.10045*, 2019.

[Avila Garcez and Zaverucha, 1999] Artur S. Avila Garcez and Gerson Zaverucha. Connectionist inductive learning and logic programming system. *Applied Intelligence*, 1999.

[Besold *et al.*, 2017] Tarek R. Besold, Artur D.Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai Uwe Kuhnberger, Luis C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo De Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. Neural-Symbolic learning and reasoning: A survey and interpretation, 2017.

[Buntine and Weigend, 1991] Wray L. Buntine and A. Weigend. Bayesian back-propagation. *Complex Syst.*, 5, 1991.

[Chen *et al.*, 2019] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. Towards knowledge-based recommender dialog system, 2019.

[Cropper *et al.*, 2020] Andrew Cropper, Sebastijan Dumančić, and Stephen H. Muggleton. Turning 30: New ideas in inductive logic programming. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4833–4839. International Joint Conferences on Artificial Intelligence Organization, 7 2020. Survey track.

[Dash *et al.*, 2018] Tirtharaj Dash, Ashwin Srinivasan, Lovekesh Vig, Oghenejokpeme I Orhobor, and Ross D King. Large-scale assessment of deep relational machines. In *International Conference on Inductive Logic Programming*, pages 22–37. Springer, 2018.

[Dash *et al.*, 2019] Tirtharaj Dash, Ashwin Srinivasan, Ramprasad S Joshi, and A Baskar. Discrete stochastic search and its application to feature-selection for deep relational machines. In *International Conference on Artificial Neural Networks*, pages 29–45. Springer, 2019.

[Dash *et al.*, 2020] Tirtharaj Dash, Ashwin Srinivasan, and Lovekesh Vig. Incorporating symbolic domain knowledge into graph neural networks, 2020.

[Demeester *et al.*, 2016] T. Demeester, Tim Rocktäschel, and S. Riedel. Lifted rule injection for relation embeddings. *ArXiv*, abs/1606.08359, 2016.

[Evans and Grefenstette, 2018] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *J. Artif. Intell. Res.*, 61:1–64, 2018.

[Fischer *et al.*, 2019] M. Fischer, Mislav Balunovic, Dana Drachsler-Cohen, Timon Gehr, Ce Zhang, and Martin T. Vechev. Dl2: Training and querying neural networks with logic. In *ICML*, 2019.

[Fletcher and Obradovic, 1993] Justin Fletcher and Zoran Obradovic. Combining prior symbolic knowledge and constructive neural network learning. *Connection Science*, 5(3-4):365–375, 1993.

[França *et al.*, 2014] Manoel VM França, Gerson Zaverucha, and Artur S d'Avila Garcez. Fast relational learning using bottom clause propositionalization with artificial neural networks. *Machine learning*, 94(1):81–104, 2014.

[Fu, 1993] L. M. Fu. Knowledge-based connectionism for revising domain theories. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(1):173–182, 1993.

[Fu, 1995] Li Min Fu. Introduction to knowledge-based neural networks. *Knowledge-Based Systems*, 1995.

[Futia and Vetrò, 2020] Giuseppe Futia and Antonio Vetrò. On the integration of knowledge graphs into deep learning models for a more comprehensible ai—three challenges for future research. *Information*, 11(2), 2020.

[Garcez *et al.*, 2012] Artur S d'Avila Garcez, Krysia B Broda, and Dov M Gabbay. *Neural-symbolic learning systems: foundations and applications*. Springer Science & Business Media, 2012.

[Hamilton *et al.*, 2018] William L. Hamilton, P. Bajaj, M. Zitnik, Dan Jurafsky, and J. Leskovec. Embedding logical queries on knowledge graphs. In *NeurIPS*, 2018.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Hogan *et al.*, 2021] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs, 2021.

[Hu *et al.*, 2016] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, E. Hovy, and E. Xing. Harnessing deep neural networks with logic rules. *ArXiv*, abs/1603.06318, 2016.

[Kitano, 2016] Hiroaki Kitano. Artificial intelligence to win the nobel prize and beyond: Creating the engine for scientific discovery. *AI magazine*, 37(1):39–49, 2016.

[Kramer *et al.*, 2001] Stefan Kramer, Nada Lavrač, and Peter Flach. *Propositionalization Approaches to Relational Data Mining*, pages 262–291. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[Krupka and Tishby, 2007] Eyal Krupka and Naftali Tishby. Incorporating prior knowledge on features into learning. In *AISTATS*, 2007.

[Kukačka *et al.*, 2017] Jan Kukačka, Vladimir Golkov, and Daniel Cremers. Regularization for deep learning: A taxonomy, 2017.

[Kursuncu *et al.*, 2019] Ugur Kursuncu, Manas Gaur, and Amit Sheth. Knowledge infused learning (k-il): Towards deep incorporation of knowledge in deep learning. *arXiv preprint arXiv:1912.00512*, 2019.

[Lavrač *et al.*, 1991] Nada Lavrač, Sašo Džeroski, and Marko Grobelnik. Learning nonrecursive definitions of relations with linus. In *European Working Session on Learning*, pages 265–281. Springer, 1991.

[Lavrac *et al.*, 2020] Nada Lavrac, Blaz Skrlj, and Marko Robnik-Sikonja. Propositionalization and embeddings: two sides of the same coin. *Mach. Learn.*, 109(7):1465–1507, 2020.

[Li and Srikumar, 2020] Tao Li and Vivek Srikumar. Augmenting neural networks with first-order logic. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2020.

[Li *et al.*, 2018] Lei Li, Min Feng, Lianwen Jin, Shenjin Chen, Lihong Ma, and Jiakai Gao. Domain knowledge embedding regularization neural networks for workload prediction and analysis in cloud computing. *J. Inf. Technol. Res.*, 11(4):137–154, October 2018.

[Lipton, 2016] Zachary C. Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.

[Liu *et al.*, 2018] Xuan Liu, Xiaoguang Wang, and Stan Matwin. Improving the interpretability of deep neural networks with knowledge distillation. *arXiv preprint arXiv:1812.10924*, 2018.

[Lodhi, 2013] Huma Lodhi. Deep relational machines. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013.

[Manhaeve *et al.*, 2018] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *arXiv preprint arXiv:1805.10872*, 2018.

[Muggleton and de Raedt, 1994] Stephen Muggleton and Luc de Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19-20:629–679, 1994. Special Issue: Ten Years of Logic Programming.

[Muggleton *et al.*, 2012] Stephen Muggleton, Luc De Raedt, David Poole, Ivan Bratko, Peter Flach, Katsumi Inoue, and Ashwin Srinivasan. Ilp turns 20. *Machine learning*, 86(1):3–23, 2012.

[Muggleton, 1991] Stephen Muggleton. Inductive logic programming. *New generation computing*, 8(4):295–318, 1991.

[Muggleton, 1995] Stephen Muggleton. Inverse entailment and progol. *New generation computing*, 13(3-4):245–286, 1995.

[Neal, 1995] Radford M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, CAN, 1995. AAINN02676.

[Park *et al.*, 2019] Namyong Park, Andrey Kan, Xin Luna Dong, Tong Zhao, and Christos Faloutsos. *Estimating Node Importance in Knowledge Graphs Using Graph Neural Networks*, page 596–606. Association for Computing Machinery, New York, NY, USA, 2019.

[Rocktäschel *et al.*, 2014] Tim Rocktäschel, Matko Bosnjak, Sameer Singh, and Sebastian Riedel. Low-dimensional embeddings of logic. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 45–49, Baltimore, MD, June 2014. Association for Computational Linguistics.

[Rocktäschel *et al.*, 2015] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[Sheth *et al.*, 2019] A. Sheth, M. Gaur, U. Kursuncu, and R. Wickramarachchi. Shades of knowledge-infused learning for enhancing deep learning. *IEEE Internet Computing*, 23(6):54–63, 2019.

[Silvestri *et al.*, 2020] Mattia Silvestri, Michele Lombardi, and Michela Milano. Injecting domain knowledge in neural networks: a controlled experiment on a constrained problem, 2020.

[Sourek *et al.*, 2018] Gustav Sourek, Vojtech Aschenbrenner, Filip Zelezný, Steven Schockaert, and Ondrej Kuzelka. Lifted relational neural networks: Efficient learning of latent relational structures. *J. Artif. Intell. Res.*, 62:69–100, 2018.

[Srinivasan *et al.*, 2019] Ashwin Srinivasan, Lovekesh Vig, and Michael Bain. Logical explanations for deep relational machines using relevance information. *Journal of Machine Learning Research*, 20(130):1–47, 2019.

[Stevens *et al.*, 2020] Rick Stevens, Valerie Taylor, Jeff Nichols, Arthur Barney Maccabe, Katherine Yelick, and David Brown. Ai for science. Technical report, Argonne National Lab.(ANL), Argonne, IL (United States), 2020.

[Takeishi and Akimoto, 2018] Naoya Takeishi and Kosuke Akimoto. Knowledge-based distant regularization in learning probabilistic models, 2018.

[Tan, 1997] Ah Hwee Tan. Cascade ARTMAP: Integrating neural computation and symbolic knowledge processing. *IEEE Transactions on Neural Networks*, 1997.

[Towell and Shavlik, 1993] Geoffrey G Towell and Jude W Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine learning*, 13(1):71–101, 1993.

[Towell and Shavlik, 1994] Geoffrey G Towell and Jude W Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.

[Towell *et al.*, 1990] Geofrey G Towell, Jude W Shavlik, and Michiel O Noordewier. Refinement of approximate domain theories by knowledge-based neural networks. In *Proceedings of the eighth National conference on Artificial intelligence*, volume 861866. Boston, MA, 1990.

[Vig *et al.*, 2017] Lovekesh Vig, Ashwin Srinivasan, Michael Bain, and Ankit Verma. An investigation into the role of domain-knowledge on the use of embeddings. In Nicolas Lachiche and Christel Vrain, editors, *Inductive Logic Programming - 27th International Conference, ILP 2017, Orléans, France, September 4-6, 2017, Revised Selected Papers*, volume 10759 of *Lecture Notes in Computer Science*, pages 169–183. Springer, 2017.

[von Rueden *et al.*, 2019] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, et al. Informed machine learning–a taxonomy and survey of integrating knowledge into learning systems. *arXiv preprint arXiv:1903.12394*, 2019.

[Wang *et al.*, 2019] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. In *The World Wide Web Conference*, WWW '19, page 3307–3313, New York, NY, USA, 2019. Association for Computing Machinery.

[Xie *et al.*, 2019] Yaqi Xie, Ziwei Xu, Mohan S. Kankanhalli, Kuldeep S. Meel, and Harold Soh. Embedding symbolic knowledge into deep networks. In *Advances in Neural Information Processing Systems*, 2019.

[Xu *et al.*, 2018] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van Den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *35th International Conference on Machine Learning, ICML 2018*, 2018.

[Yosinski *et al.*, 2014] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, pages 3320–3328, 2014.

[Zhuang *et al.*, 2020] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning, 2020.