First International Workshop on Capturing Scientific Knowledge (SciKnow-15), Collocated with the Eighth ACM International Conference on Knowledge Capture (K-CAP), Palisades, NY, October 2015.

Exploring Synergies between Machine Learning and Knowledge Representation to Capture Scientific Knowledge

Imme Ebert-Uphoff
School of Electrical and Computer Engineering
Colorado State University
Fort Collins, CO, USA
iebert@engr.colostate.edu

Yolanda Gil
Information Sciences Institute and
Department of Computer Science
University of Southern California
Marina del Rey, CA 90292, USA
gil@isi.edu

ABSTRACT

In this paper we explore synergies between the machine learning and knowledge representation fields by considering how scientific knowledge is represented in these areas. We illustrate some of the knowledge obtained through machine learning methods, providing two contrasting examples of such models: probabilistic graphical models (aka Bayesian networks) and artificial neural networks (including deep learning networks). From knowledge representation, we give an overview of ontological representations, qualitative reasoning, and planning. Then we discuss potential synergies that would benefit both areas.

Categories and Subject Descriptors
I.2.4 [**Knowledge Representation Formalisms and Methods**]: Representation languages.

General Terms
Languages.

Keywords
Machine learning, knowledge representation, scientific knowledge.

1. INTRODUCTION

Scientists from many disciplines like to describe domain knowledge in terms of *models* that capture the most important relationships governing a phenomenon or system under consideration. For example, mathematical models, such as differential equations, describe the dynamics of systems in disciplines ranging from engineering to natural sciences (e.g. physics, biology and earth sciences) and social sciences (e.g. economics, sociology and psychology). Thus, by definition, such *models* encode expert knowledge about a domain.

Researchers in both statistics and machine learning have developed numerous techniques for the process of *system identification*, which is the process of deriving a model from observed data of a system, to help scientists generate or refine existing models. Great advances are resulting from machine learning in science, particularly in bioinformatics, economics, social sciences, ecology and climate science.

But learning from data alone is often not the most efficient means when studying complex phenomena. Instead it is often helpful to include prior knowledge. Advanced knowledge representation techniques that capture important structural and process characteristics could provide valuable domain knowledge to machine learning algorithms, and have a significant impact on our ability to understand complex scientific phenomena.

This paper explores possible synergies between machine learning and knowledge representation approaches to capturing scientific knowledge. To provide the reader with an intuitive understanding of the type of scientific knowledge that can be gained through machine learning, we first discuss two sample machine learning methods. Then we give an overview of the capabilities of modern knowledge representation systems. We explore synergies between these areas through a discussion on how machine learning models can be integrated in existing knowledge representation frameworks, and vice versa.

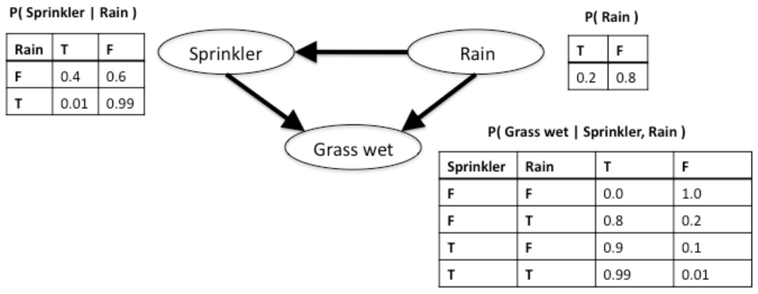
2. MACHINE LEARNING MODELS

Machine learning models are simply models learned automatically, primarily from sample data of the studied system. The majority of those models are developed for specific tasks, such as prediction or classification, while some are developed specifically for system identification.

In this context we distinguish two different types of machine learning models:

(1) *Black box models* are models that only seek to mimic the input-output relationships of the real-world system without caring about the internal relationships. In other words, black box models aim at, given a certain input to the system, yielding the same output of the system as the real-world system would. Classic regression models are a typical example.

(2) *Clear box models* (also called *white box* or *glass box models*) seek to also internally mimic the key relationships inside the real-world system as truthfully as possible. An example is any model obtained using causal discovery



P(Sprinkler | Rain)

Rain	T	F
F	0.4	0.6
T	0.01	0.99

P(Rain)

T	F
0.2	0.8

P(Grass wet | Sprinkler, Rain)

Sprinkler	Rain	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Figure 1. The classic sprinkler example is one of the simplest probabilistic graphical models.

methods, e.g. certain probabilistic graphical models trained from data.

In practice the distinction between black and clear box models is not crisp, i.e. models are more on a continuum. In fact, many models that are meant to be only black box models end up encoding information about some of the internal structure of the system. To make this discussion more concrete, this section provides two examples of machine learning models that can be interpreted as representing scientific knowledge. We chose two very different methods, namely probabilistic graphical models - that are intended to be clear box models - and artificial neural networks - that were originally intended as black box models.

2.1 Probabilistic Graphical Models

The type of machine learning model that is probably most closely related to standard methods of knowledge representation is a probabilistic graphical model. Indeed, probabilistic graphical models (or graphical models for short) are often described as ontologies supplemented with probabilities. See [1] for a basic introduction to the topic, or [2,3] for more detailed information.

Figure 1 demonstrates a very simple graphical model, namely the classic sprinkler model. The sprinkler model encodes the relationships between three different variables of a lawn: *Sprinkler* represents whether the sprinkler system has recently been activated, *Rain* represents whether it recently rained in the neighborhood of the lawn, and *Grass wet* represents whether the grass is wet. The relationships in this model indicate that whether it rained has an influence on whether the sprinkler system was activated, i.e. the sprinkler may have manually been turned off if there was recent rain. Furthermore, whether the grass may be wet is affected by whether the sprinkler recently ran and whether there was a recent rain. The *graphical structure* of the model represents these cause-effect relationships, where the arrows always go in the direction from cause to effect. All three variables in this simple example are assumed to be binary, i.e. each can only take the values true (T) or false (F). The tables next to each variable in Figure 1 provide the probabilities for the occurrence of its states, based on the states of all of its parents. For example, the probability that the grass is detected as wet, if it is known that the sprinkler was turned off and that it recently rained in the neighborhood, is 0.8.

While the sprinkler system provides a tiny example, a graphical model may have 100s or even 1,000s of nodes. The type of graphical model shown in Figure 1, where all edges are directed, is known as a *Bayesian network*. An undirected model is known as a *Markov network.*

2.1.1 Learning graphical models

Graphical models can be generated completely from expert knowledge - that is often done in the medical field by asking medical professionals. They can also be learned completely from sample data - that is often done in bioinformatics and recently in climate science. Often a combination of both is used, i.e. learning the model from data while constraining the learning based on expert knowledge.

Learning such a model involves two major parts, namely (1) learning the graph and (2) learning the probabilities. The former part, namely learning the graph structure, is known as *structure learning.* The latter part, namely learning the probabilities, is known as *parameter learning*. Structure learning is a very complex problem - especially given the fact that the potential existence of hidden common causes has to be taken into account [2,3], and that the computational complexity considering 100s and 1,000s of variables is very high. Nevertheless, structure learning has been applied successfully and generated new knowledge in many applications with a large number of variables, for example in bioinformatics to identify protein-protein interactions and gene regulatory networks [4]. In climate science they have been used to identify pathways of interactions of dynamical processes around the globe [5]. In contrast, parameter learning, i.e. learning the probabilities of the model once the graph structure is known, is a comparatively easy task.

2.1.2 Inference with graphical models

Bayesian networks are often used for the purpose of inference. For example, if we know the states of certain variables, the model allows us to enter those states, then propagates them through the model so that the probabilities of all other variables can be inferred. For example, for the sprinkler system one may ask, if we know that the grass is currently wet, what is the probability that the sprinkler

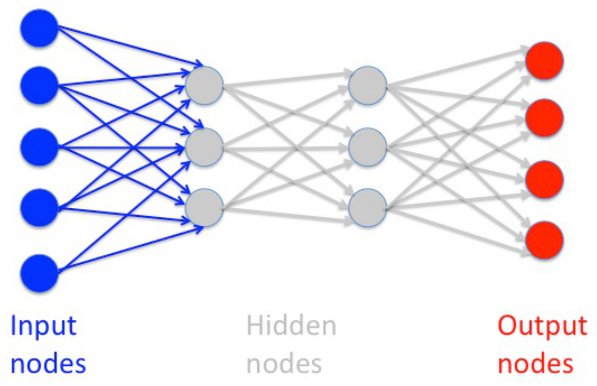


Figure 2: Artificial neural network (ANN).

recently ran or the probability that it recently rained? Such inferences are particularly helpful for example for medical diagnosis. Given that we see certain symptoms in a patient, which disease included in the model is the most likely cause?

However, an entirely different use of graphical models, which is often referred to as *causal discovery*, has recently emerged and is gaining popularity. The idea of causal discovery is that we use structure learning *to learn the graph model from data*, and the resulting graph (with or without probabilities) is the end result of the learning process. Namely, rather than using the model *as a tool* for inference, we *study the model itself* to learn about the potential cause-effect relationships that were identified [3,4,5].

In summary, causal discovery seeks to identify the potential cause-effect relationships between variables of a system under consideration from observations, and to encode them in a graph representation of the type shown in Figure 1. This graph representation is a form of knowledge representation that captures scientific knowledge.

2.2 Artificial Neural Networks and Deep Learning

The key idea behind neural networks - of which deep learning networks are a special case - is to mimic the way the human brain works. The brain consists of a network of neurons that are connected to each other through signal pathways. Artificial neural networks (ANNs) seek to mimic this type of structure, by modeling each neuron as a node of a simulated network, but in this artificial version all neurons are arranged in layers. Each connection in the network has a weight assigned to it that indicates the strength of the connection and is learned from data during an initial training phase.

There are three types of nodes, *input* nodes, *hidden* nodes and *output* nodes, as shown in Fig. 2. The input nodes are in the first layer (on the left in Fig. 2), followed by one or more layers of hidden nodes (center in Fig. 2) and finally one layer of output nodes. Each input node takes in information from an external source. For example if the network is used for image processing each input node may represent the intensity of exactly one pixel in an image.

The value that any node takes is called its *activation level*. The values of the input nodes are then passed on to the next level (to the right) through the connections in the network, where the amount of signal being passed on depends on the connection weights. The activation levels are passed through those weighted pathways from layer to layer until they reach the output layer. The activation levels of the output nodes then represent the output of the model. Only the nodes in the input and output layers have a predefined meaning, while all the nodes in the hidden layers are meant to have no specific (pre-conceived) meaning, just like the majority of neurons in the human brain do not start out with a specific meaning before they are being trained to serve some specific function.

Obviously, this is a very simple model compared to the incredible complexity of the human brain, but it has led to very interesting results and insights. Furthermore, ANNs are very useful because they yield statistical approximation methods that are both robust and hierarchical. New methods for training such networks developed in the past ten years have made it possible to train networks with a very large number of hidden layers, known as *deep learning networks* (or *deep networks* for short). Deep networks are a hot topic in machine learning, because they can achieve superior results for tasks such as image recognition, speech recognition and natural language processing.

2.2.1 Example: Recognizing written digits

This section illustrates artificial neural networks using an example provided by Michael Nielsen in his free online book [6]. Figures 3 to 6 in this section are all from [6] (reuse is permitted under Creative Commons Attribution-NonCommercial 3.0 Unported License, see https://creativecommons. org/licenses/by-nc/3.0/us/.) As shown in Figure 3, the ANN for this application uses only one hidden layer. This particular network was designed to recognize a single hand-written digit, from an image that is 28x28 pixels, i.e. 784 pixels. The values for each pixel are fed, one by one, into the input layer of the network (thus there are 784 neurons in the input layer). Figure 4 provides an example of such hand-written images.

The output layer of the network contains 10 different neurons, one for each digit that might occur (numbers 0 to 9). The network is trained to minimize the number of errors made at the output layer. Details can be found in [6]. Once the network is trained to recognize the digits, we feed the 28x28 image of a digit to be recognized into the 784 input neurons, then check which of the output neurons has the highest activation level, indicating which digit the system believes is most likely encoded in the image.

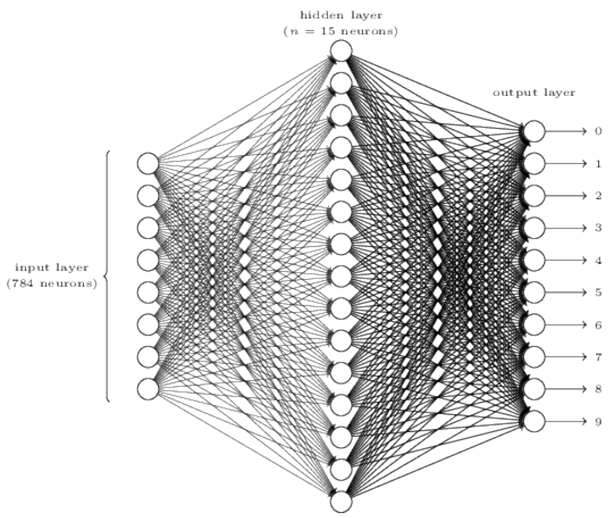


Figure 3: Three-layer neural network with 784 neurons in input layer, 15 neurons in hidden layer and 10 neurons in output layer. Image source: [6].

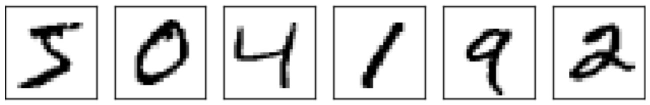


Figure 4: Example of individual hand-written digits to be recognized by neural network. Image source: [6].

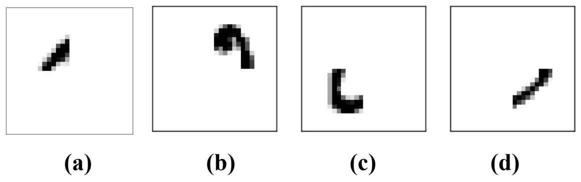


Figure 5: Partial image patterns. Image source: [6].

How does the system achieve this task? And what role does the hidden layer of neurons play? It turns out that neurons in the hidden layer may look for certain *combinations* of input pixels to be present in the image. For example, to recognize the number zero, one neuron in the hidden layer may look for the presence of a pattern like the one in Figure 5(a), while other neurons may be looking for the other patterns in Figure 5. The neurons in the hidden layer achieve that functionality by weighing pixels that overlap with the partial pattern they are responsible for. These four image patterns together allow us to recognize the digit 0 (Fig. 6). In other words, if the four neurons in the hidden layer responsible for detecting the partial image patterns above all have high activation levels, then - due to the connectivity of the network achieved through training - the neuron in the output layer that represents the digit 0 is highly activated.

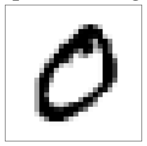


Figure 6: Digit zero.

2.2.2 Neural networks are no longer black boxes

Neural networks were designed to act as *black box models*, i.e. the meaning of the nodes in the hidden layers was often interpreted to be of little relevance. This has changed drastically especially with the advent of deep learning networks. Study of the hidden nodes revealed that with each additional layer, the model is able to achieve higher layers of abstraction. The first layers can only represent some very basic properties (such as the presence of partial image patterns such as in Figures 5), while consecutive layers can build on the properties encoded in the first layers to encode properties of increasingly higher abstraction. Studying how the model chooses to represent the data reveals much about the internal properties of the data. For this example the system was able to decompose the input data into basic building blocks, such as the ones in Figures 5 and 6. The key point is that *the network chose the building blocks all by itself*. Similarly, if instead of images of hand-written digits we provide only images of faces to a similar network, the network decomposes the face images into building blocks, such as ears, eyes, noses, etc., that make it possible for the system to recognize whether an image is a face [7]. More importantly, we can learn from the building blocks what important components a human face consists of. For a human face, the resulting building blocks are not really surprising and not that new, since face recognition is a task that our brains are already well trained in. However, our brains clearly are not well trained at looking at observed data from complex natural systems and finding patterns - so what patterns (building blocks) would the network find there? How do those patterns help us understand the system being studied? Imagine how much better we could understand natural systems, say ocean currents or the formation of tornadoes, if computers could help us gain an intuitive understanding of the observed data by decomposing it into the equivalent of ears, eyes and noses?

Thus, although originally designed as a black box model, we can now learn from the ANN model itself, which means that this type of ANN model itself represents new insights that can lead to new scientific knowledge of the system, by recognizing building blocks or important features, as proposed for example for climate data in [8].

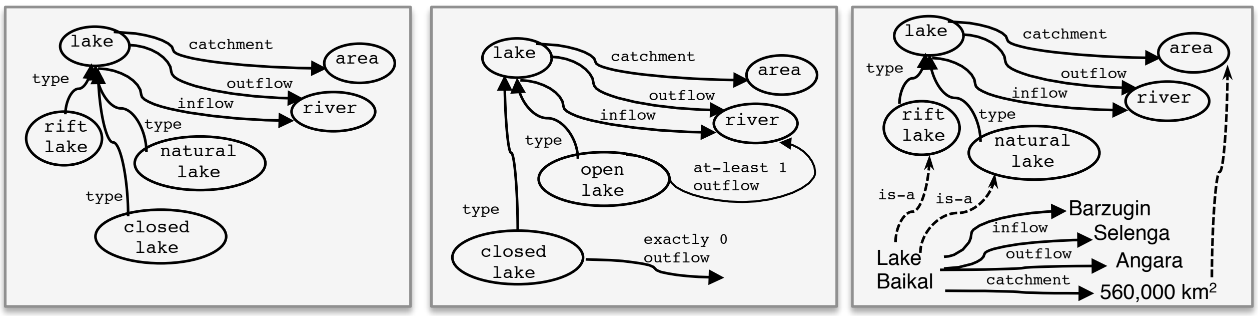
Although we only discussed two methods here, there are many other types of machine learning methods that encapsulate scientific knowledge each in their own way.

3. KNOWLEDGE REPRESENTATION

Many different models are used in the field of knowledge representation. Most models are based on first-order logic, either as a subset or as an extension of it.

3.1 First-Order Logic

Some researchers favor first-order logic because of its expressive power and inference capabilities. Recent work on Markov logic networks [9] combines first-order logic



(a) Classes and their properties (b) Class definitions based on constraints (c) Assertions about Objects

Figure 7. Ontologies to represent descriptive and factual knowledge: (a) classes and subclasses can be defined, each with their properties; (b) some classes can have definitions based on constraints on their properties; (c) assertions can be made about objects and their properties.

and probabilistic graphical models, and could be a bridge between machine learning and knowledge representation work.

3.2 Ontologies

Ontologies offer a small subset of the expressivity of first order logic that is computationally tractable. They are widely adopted in many sciences, notably in biomedical research.

Figure 7 illustrates how knowledge is represented in ontologies. Many ontologies simply represent classes and their properties, as in traditional AI frame systems. In Figure 7(a), the class "lake" has several subclasses (rift, closed, and natural lakes), and also several properties (catchment, outflow, and inflow). Properties are constrained by what values they can take. In this example, the outflow must be a river. Classes can have definitions based on constraints on their properties. Figure 7(b) shows an example, where an open lake is defined as a type of lake that has at least one outflow, and a closed lake is a lake that has no outflow. These classes and properties are used to describe objects. Shown in Figure 7(c) are several assertions about the object Lake Baikal, for example that it has inflow from Barzugin and Selenga and has an area of catchment of 560,000 km^2.

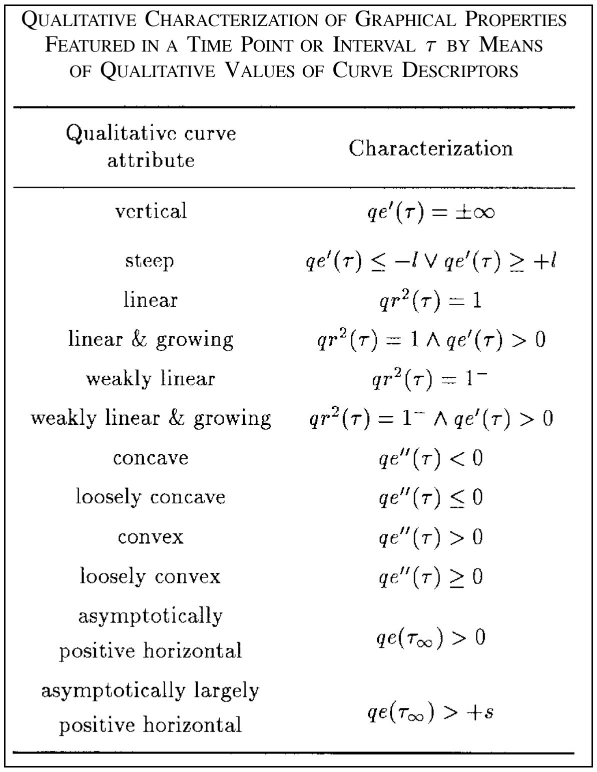
Ontological reasoners can make inferences on these representations. An example of inference is inheritance, where the properties of a class are applied to all its subclasses and any asserted objects of that type. Inferences can be made about the classes of new objects asserted in the system. In the examples in Figure 7, since the inflow and outflow of lakes are to rivers, the system can infer that Barzugin, Selenga, and Angara are all rivers. These reasoners can also be used to answer questions about rivers that flow into Lake Baikal. A major use of ontologies is to ensure that all the data about objects is consistent with what is expected of the world. So if someone indicated that Lake Baikal is a closed lake then the system would indicate it as an inconsistency.

The most popular languages to represent ontological knowledge are used to represent knowledge on the web. The Resource Description Framework (RDF) [10] provides a simple frame-like representation with very limited inference capabilities. The Web Ontology Language (OWL) offers the representation and inference capabilities of a description logic, similar to those illustrated in Figure 9. A major shift in ontological knowledge representation in the last decade has been that all the concepts and objects in knowledge bases are Web objects and openly accessible from a URI. As a result, they can be easily reused and mixed with the contents of other sources to create new knowledge bases. Massive amounts of knowledge are available in this form, and known as Linked Open Data.

3.3 Process Representations

Scientific knowledge includes descriptions of the behaviors of systems or phenomena that change over time.

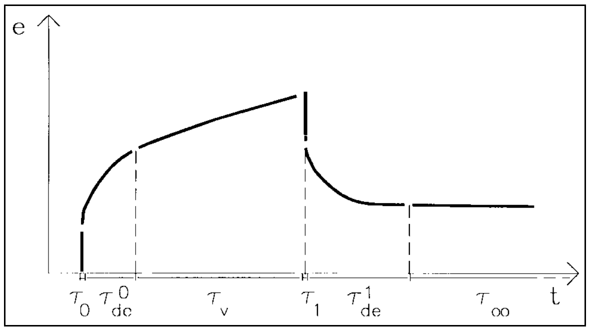
Qualitative representations are qualitative abstractions of quantitative data about a system. For physical processes, ordinary differential equations are often used to describe a system quantitatively. Figure 8 shows an example of abstractions to represent a system qualitatively (taken from [11]), where some qualitative attributes of a curve can be used to turn detailed quantitative data in a curve about material deformation and creep into a set of logic rules to assess elasticity properties. These qualitative representations are used for simulation as well as for causal reasoning. Qualitative representations reduce the computational complexity over quantitative models, enable explanation generation, and facilitate exploration of what-if scenarios. Special treatment is given to spatial and temporal abstractions, as well as descriptions of states, state transitions, and behaviors. A crucial issue is the composition of qualitative reasoning models to enable the



QUALITATIVE CHARACTERIZATION OF GRAPHICAL PROPERTIES FEATURED IN A TIME POINT OR INTERVAL τ BY MEANS OF QUALITATIVE VALUES OF CURVE DESCRIPTORS

Qualitative curve attribute	Characterization
vertical	$qe'(\tau) = \pm\infty$
steep	$qe'(\tau) \leq -l \vee qe'(\tau) \geq +l$
linear	$qr^2(\tau) = 1$
linear & growing	$qr^2(\tau) = 1 \wedge qe'(\tau) > 0$
weakly linear	$qr^2(\tau) = 1^-$
weakly linear & growing	$qr^2(\tau) = 1^- \wedge qe'(\tau) > 0$
concave	$qe''(\tau) < 0$
loosely concave	$qe''(\tau) \leq 0$
convex	$qe''(\tau) > 0$
loosely convex	$qe''(\tau) \geq 0$
asymptotically positive horizontal	$qe(\tau_\infty) > 0$
asymptotically largely positive horizontal	$qe(\tau_\infty) > +s$

(a)



(b)

```
if curve is
  vertical at $\tau_0$ AND steep at $\tau_1$
  OR steep at $\tau_0$ AND vertical at $\tau_1$
then
  instantaneous-elasticity property holds
True
else
  instantaneous-elasticity property holds
False
endif
```

(c)

Figure 8. Qualitative reasoning about processes involves creating abstractions about a system under and using them to reason about how it behaves. Shown here are examples by [11] from materials science, with abstractions shown in (a), how the abstractions are used to describe the actual behavior of the system, in this case creep deformation of a material (b), and how the qualitative abstractions are used in logic rules to reason about the system (c).

```
(:durative-action heat-water
   :parameters (?p - pan)
   :duration (= ?duration (/ (- 100 (temperature ?p)) (heat-rate)))
   :condition (and (at start (full ?p))
                   (at start (onHeatSource ?p))
                   (at start (byPan))
                   (over all (full ?p))
                   (over all (onHeatSource ?p))
                   (over all (heating ?p))
                   (at end (byPan)))
   :effect (and    (at start (heating ?p))
                   (at end (not (heating ?p)))
                   (at end (assign (temperature ?p) 100)))
)
```

Figure 9. A representation for the process of heating water as an action with conditions and effects (from [McDermott 2000].

treatment of a physical system as a "system of systems". An introductory overview is provided in [12]. Qualitative representations are often used in combination with ontologies that describe system components or situations.

AI planning approaches represent processes by decomposing them into individual actions with preconditions and effects. Figure 9 shows an example of an action for boiling water taken from [13]. The conditions and effects can be used to generate plans composed of actions given an initial state and a desired goal.

A number of additional formalisms have been proposed by knowledge representation researchers to address state changes, resource use, temporal reasoning, and other aspects of reasoning about actions and plans.

4. EXPLORING SYNERGIES

This article aims to stimulate discussion and cooperation between the machine learning and knowledge representation communities in representing scientific knowledge, by leveraging and merging existing techniques from both disciplines.

4.1 Existing Work

There are several areas in which such cooperation has already taken place. For example, *Markov logic networks (MLNs)* combine the frameworks of first order-logic and probabilistic graphical models [9] in order to add a probabilistic component to knowledge bases. Namely, the formulas of a first-order knowledge base imposes *hard constraints* on the set of possible worlds - if a world violates even one formula, it has *zero* probability. MLNs add weights to these formulas to *soften* these constraints - as a result a world that violates one formula is just *less probable*. Similarly, *knowledge-based model construction (KBMC)* combines logic programming and Bayesian networks [14], and is a precursor of MLNs and can in fact be seen as a special case of MLNs [9]. [9] also discusses several other related approaches, including *probabilistic relational models*, *relational Markov networks* and other *logic programming* approaches.

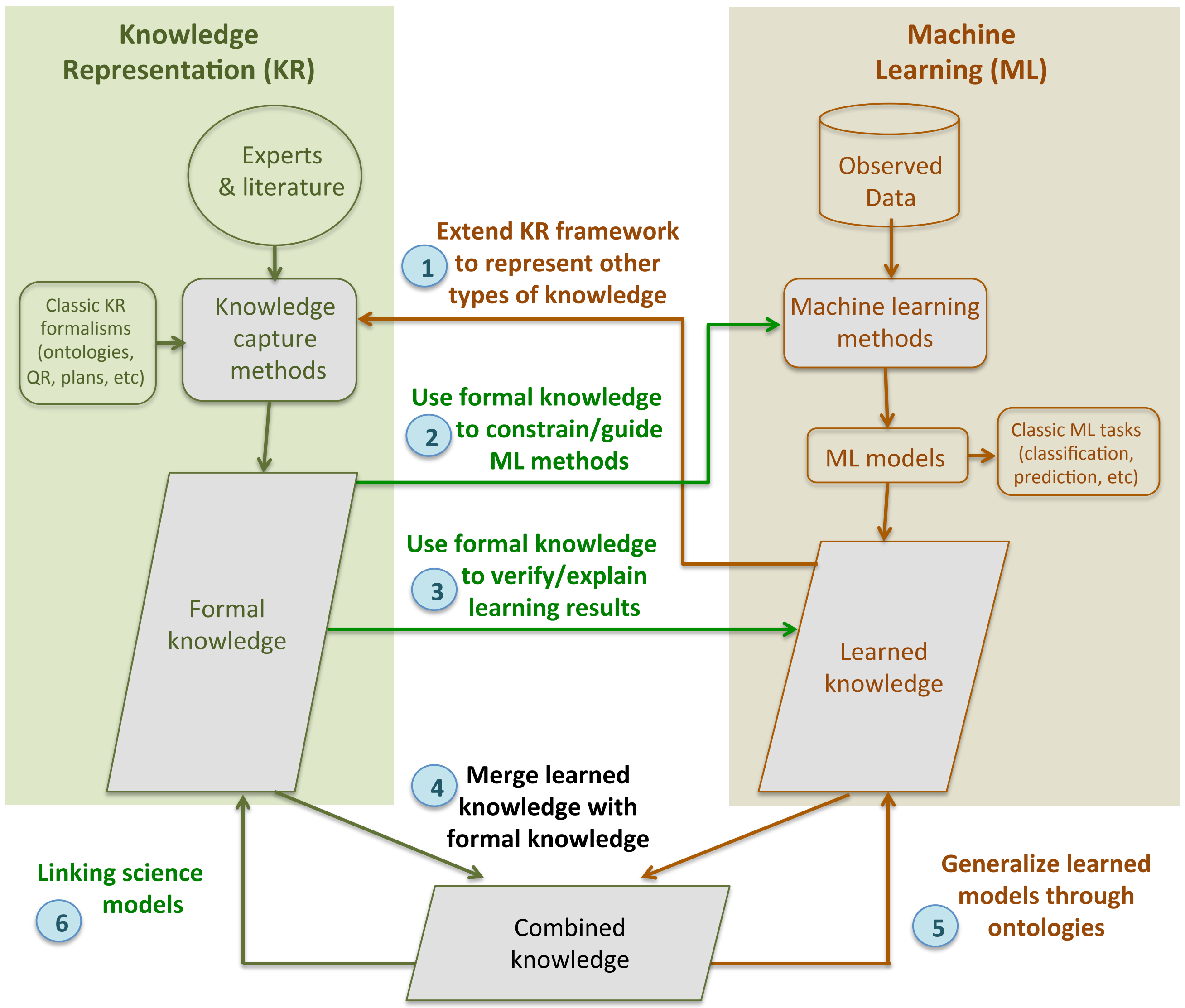

Knowledge Representation (KR)
Machine Learning (ML)
Experts & literature
Observed Data
Classic KR formalisms (ontologies, QR, plans, etc)
Knowledge capture methods
Machine learning methods
ML models
Classic ML tasks (classification, prediction, etc)
Formal knowledge
Learned knowledge
Combined knowledge
1 Extend KR framework to represent other types of knowledge
2 Use formal knowledge to constrain/guide ML methods
3 Use formal knowledge to verify/explain learning results
4 Merge learned knowledge with formal knowledge
5 Generalize learned models through ontologies
6 Linking science models

Figure 10: Overview of potential synergistic interactions between KR and ML research

ANNs have been used successfully for reasoning in knowledge bases. For example, Bordes et al. [15] and Socher et al. [16], seek to infer new relations between entities from existing relationships between other entities in the database. Both use variations of ANNs for that purpose. Bordes et al. [15] propose their method to transfer knowledge *between different databases* and to *integrate the database knowledge into machine learning methods*. Thus in the latter case, machine learning becomes both the *means* of integration (in this case ANNs), as well as the *target* of the integration. Another approach is *Knowledge-Based Artificial Neural Networks (KBANN)* [17], which map domain knowledge represented in propositional logic into ANNs. In KBANN, the assertions are used to create the structure of the network, and the learning algorithms extend the network and adjust the weights based on the examples.

4.2 Additional Synergistic Research Areas

Fig. 10 provides a general overview of potential synergistic interactions between KR and ML research. The KR box on the left indicates the standard process of KR leading from *experts and literature* to *formal knowledge*. The ML box on the right indicates the standard process of ML leading from *observed data* of a system under consideration to a model from which knowledge can be extracted. However, this type of knowledge, which we in the following refer to as *learned knowledge,* is often not represented in a formal way that would be easy to use for outsiders.

Fig. 10 lists six types of potential interactions (1-6), which are discussed below. Some of them point out how KR research can be useful for ML research. Others point out how ML models can benefit KR research. The last two mentioned would open the way to fundamentally new capabilities resulting from synergistic research.

4.2.1 Extending KR Frameworks to Represent ML Results

In order for learned knowledge to be useful for scientific discovery, it must be represented in a way that is easily accessible and understandable for domain experts. For example, the specific parameters of a ML model have no meaning for someone who is not an expert in the particular ML method used, but that information also cannot be expressed in first-order logic and its derivatives. Thus we need to find ways to translate that type of information into information meaningful to a domain expert. This requires (1) significant research on the ML side, namely to develop methods for extraction of the information for each type of ML technique; and (2) significant collaboration with KR researchers to develop formalisms that capture the knowledge and are easy to understand by domain experts. Basically, a new language must be developed for ML methods, that is meaningful and accessible to domain experts. Markov logic networks [9] provide an example of such research, as they extend the KR framework to include the probabilistic component inherent in probabilistic graphical models. Many more such extensions are needed.

4.2.2 Using KR to Guide ML Models

Formal knowledge captured by KR methods represents a wealth of expert knowledge for many domains. Many ML methods are able to use such information as constraints while building the models, but currently there seems to be little interaction between the two areas. Suggestions: One could use information about a domain contained in ontologies to select which variables to include in a graphical model or neural network. Furthermore, one can constrain structure learning of graphical models by providing a list of forced and forbidden connections, obtained for example from ontologies. In cases where domain knowledge in KR is particularly rich, one could even extract initial structures from domain ontologies and use them as initial guess to be refined by structure learning of graphical models. Likewise, many other ML methods allow to enter prior knowledge to constrain the learning process. It seems that a major hurdle is simply that the ML community is not fully aware of the domain knowledge available through KR, and that the KR community is not fully aware of the things the ML community could do with that knowledge.

A specific research area that would have high impact is to enrich learning models with spatio-temporal reasoning. A variety of spatial and temporal representations have been investigated in knowledge representation, which could be exploited by machine learning algorithms to frame learning based on a spatio-temporal basis.

4.2.3 Explaining and Verifying ML Models with KR

After a model is learned using machine learning methods - with or without the use of KR information - we can use formal knowledge to verify the results. For example, a graphical model could be mapped into a domain ontology in order to check whether its variables or components have a counterpart in the ontology and the inferences made are sound and consistent with the model. The classes and properties in an ontology about the particular domain might yield hints for the interpretation of hidden nodes in graphical models or neural networks.

A related use of KR information is to verify the results of a learned model, where the lack of an explanation would indicate potential issues in learned models. For example, a graphical model could be mapped into a domain ontology in order to check whether its variables or components have a counterpart in the ontology and the inferences made are sound and consistent with the model.

4.2.4 Merging Learned Knowledge with Formal Knowledge

Just like formal knowledge can be used to guide ML methods, ML results can also be used to inform ontologies. Ontologies could be augmented with concepts and properties formulated from components of learned models. A learned model exposes new variables in graphical models or meanings of hidden nodes in neural networks, which could be used to propose extensions to domain ontologies. In addition, learned models ground the knowledge with data, providing context for the knowledge in terms of situations and hypotheses.

Another possibility would be to create ontologies from the structures of learned models. Graphical models or neural networks could be used to generate new ontologies to represent knowledge in new areas, or to bridge across existing ontologies that are partially covered by the training data.

4.2.5 Generalizing Learned Models Through Formal Knowledge

Ontologies include hierarchies of classes of objects in the domain and their associated common properties, which could be used as a generalization bias by machine learning algorithms. This could help in generalizing learned models, particularly when few examples are available.

4.2.6 Linking Learned Models

Different learned models could be mapped to different subsets of an ontology (or of several mapped ones). Then, the ontology could be used to propose mappings that link together the learned models. This could enable new forms of multi-model learning, where a different algorithm could be used to learn from different aspects of the data, and the learned models would then be integrated together.

5. CONCLUSIONS

There are significant potential synergies between the areas of machine learning and knowledge representation. Knowledge representation researchers may find a new area of interest in studying machine learning representations and integrating them into existing frameworks. Machine learning experts may learn new ways to leverage their models by incorporating knowledge representation techniques. Novel research could be enabled through these synergistic areas of work.

6. ACKNOWLEDGMENTS

The authors would like to thank the participants of the NSF Workshop on Intelligent Systems for Geosciences, supported by NSF award # IIS-1533930. Support for this work was provided to Ebert-Uphoff by the NSF Climate and Large-Scale Dynamics (CLD) program under Grant AGS-1445978. The first author would like to thank Charles Anderson (Computer Science, Colorado State University) for recently getting her interested in the topic of neural networks and their applications to climate science.

7. REFERENCES

[1] Charniak, E., 1991: Bayesian networks without tears. AI Mag., 12 (4), 1991, pp. 50–63.

[2] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Net- works of Plausible Inference, 2nd ed., Morgan Kaufman Publishers, 552 pp, 1988.

[3] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, Springer Lecture Notes in Statistics. 1st ed. Springer Verlag, 526 pp., 1993.

[4] N. Friedman, M. Linial, I. Nachman and D. Peer, "Using Bayesian networks to analyze expression data", J. Comput. Biol., 7 (34), 601620, 2000.

[5] I. Ebert-Uphoff and Y. Deng, "A New Type of Climate Network based on Probabilistic Graphical Models: Results of Boreal Winter versus Summer", Geophysical Research Letters, vol. 39, L19701, 7 pages, 2012.

[6] Nielson, Michael, "Neural Networks and Deep Learning", Determination Press, 2015. Free online book available at http://neuralnetworksanddeeplearning.com.

[7] Jones, Nicola, "The Learning Machines", *Nature*, Vol. 505, Jan. 2014, pp. 146-148.

[8] C. Anderson, I. Ebert-Uphoff, Y. Deng and M. Ryan, "Discovering Spatial and Temporal Patterns in Climate Data Using Deep Learning", 5th International Workshop on Climate Informatics, NCAR Mesa lab, Boulder, CO, Sept 2015.

[9] M. Richardson and P. Domingos, "Markov Logic Networks", Machine Learning, vol. 62, issue 1, pp. 107-136, Feb 2006.

[10] Dan Brickley and R.V. Guha. 2004. "RDF Vocabulary Description Language 1.0: RDF Schema." World Wide Web Consortium. Available from http://www.w3.org/TR/rdf-schema.

[11] Capello, A.C. Ironi, L. and Tentoni, S. "Automated mathematical modeling from experimental data: an application to material science." IEEE Transactions on Systems, Man, and Cybernetics, Vol 28 Issue 3, 1998. DOI: 10.1109/5326.704564

[12] Kenneth D. Forbus. "Qualitative Reasoning". In CRC Handbook of Computer Science, 1996. Available from http://www.qrg.northwestern.edu/papers/files/crc7.pdf

[13] Drew M. McDermott. "The 1998 AI Planning Systems Competition." AI Magazine, Vol 21, No 2, 2000.

[14] K. Kersting and L. De Raedt, "Towards Combining Inductive Logic Programming with Bayesian Networks", Inductive Logic Programming, Lecture Notes in Computer Science, Vol. 2157, Springer, pp 118-131, 2001.

[15] A. Bordes, J. Weston, R. Collobert, Y. Bengio, "Learning Structured Embeddings of Knowledge Bases", Proceedings of the 25 the Conference on Artificial Intelligence (AAAI-11), San Francisco, CA, Aug 2011.

[16] R. Socher, D. Chen, C.D. Manning, A. Ng, "Reasoning With Neural Tensor Networks for Knowledge Base Completion", Advances in Neural Information Processing Systems (NIPS 2013), pp. 926--934, Dec 2013.

[17] G.G. Towell and J.W. Shavlik, "Knowledge-based artificial neural networks", Artificial Intelligence, vol. 70. no. 1, pp. 119-165, 1994.