DEEP ADAPTIVE SEMANTIC LOGIC (DASL): COMPILING DECLARATIVE KNOWLEDGE INTO DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce **Deep Adaptive Semantic Logic (DASL)**, a novel framework for automating the generation of deep neural networks that incorporates user-provided formal knowledge to improve learning from data. We provide formal semantics that demonstrate that our knowledge representation captures all of first order logic and that finite sampling from infinite domains converges to correct truth values. DASL's representation improves on prior neuro-symbolic work by avoiding vanishing gradients, allowing deeper logical structure, and enabling richer interactions between the knowledge and learning components. We illustrate DASL through a toy problem in which we add structure to an image classification task and demonstrate that knowledge of that structure reduces data requirements by a factor of 1000. We apply DASL on a visual relationship detection task and demonstrate that the addition of commonsense knowledge improves performance by 10.7% in a data scarce setting.

1 INTRODUCTION

Early work on Artificial Intelligence focused on Knowledge Representation and Reasoning (KRR) through the application of techniques from mathematical logic Genesereth & Nilsson (1987). The compositionality of KRR techniques provides expressive power for capturing expert knowledge in the form of rules or assertions (*declarative knowledge*), but they are brittle and unable to generalize or scale. Recent work has focused on Deep Learning (DL), in which the parameters of complex functions are estimated from data LeCun et al. (2015). DL techniques learn to recognize patterns not easily captured by rules and generalize well from data, but they often require large amounts of data for learning and in most cases do not reason at all Yang et al. (2017); Garcez et al. (2012); Marcus (2018); Weiss et al. (2016). In this paper we present **Deep Adaptive Semantic Logic (DASL)**, a framework that attempts to take advantage of the complementary strengths of KRR and DL by fitting a model simultaneously to data and declarative knowledge. DASL enables robust abstract reasoning and application of domain knowledge to reduce data requirements and control model generalization.

DASL represents declarative knowledge as assertions in first order logic. The relations and functions that make up the vocabulary of the domain are implemented by neural networks that can have arbitrary structure. The logical connectives in the assertions compose these networks into a single deep network that is trained to maximize their truth. Figure 1 provides an example network that implements a simple rule set through composition of network components performing image classification. Logical quantifiers "for all" and "there exists" generate subsamples of the data on which the network is trained. DASL treats labels like assertions about data, removing any distinction between knowledge and data. This provides a mechanism by which supervised, semi-supervised, unsupervised, and distantly supervised learning can take place simultaneously in a single network under a single training regime.

The field of neuro-symbolic computing Garcez et al. (2019) focuses on combining logical and neural network techniques in general, and the approach of Serafini & Garcez (2016) may be the closest of any prior work to DASL. To generate differentiable functions to support backpropagation, these approaches replace pure Boolean values of 0 and 1 for *True* and *False* with continuous values from $[0, 1]$ and select fuzzy logic operators for implementing the Boolean connectives. These operators generally employ maximum or minimum functions, removing all gradient information at the limits,

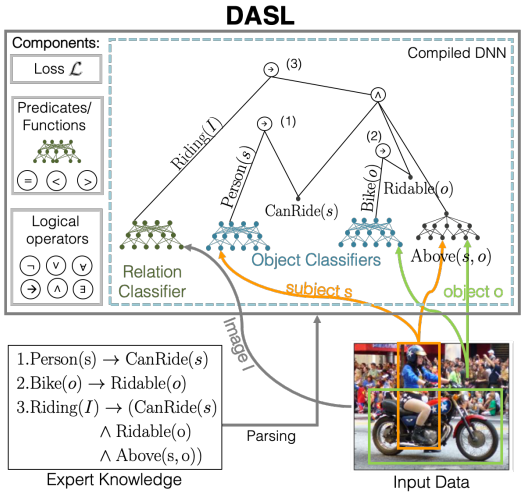

Figure 1: DASL integrates user-provided expert knowledge with training data to learn DNNs. It achieves this by compiling a DNN from knowledge, expressed in first order logic, and domain-specific neural components. This DNN is trained using backpropagation, fitting both the data and knowledge. Here DASL applies commonsense knowledge to the visual relationship detection task. $\wedge$ and $\rightarrow$ refer to 'and' and 'implies' connectives respectively.

or else they use a product, which drives derivatives toward 0 so that there is very little gradient for learning (see section A.7 in supplementary). DASL circumvents these issues by using a logit representation of truth values, for which the range is all real numbers.

Approaches to knowledge representation, both in classical AI and in neuro-symbolic computing, often restrict the language to fragments of first order logic (FOL) in order to reduce computational complexity. We demonstrate that DASL captures full FOL with arbitrary nested quantifiers, function symbols, and equality by providing a single formal semantics that unifies DASL models with classical Tarski-style model theory Chang & Keisler (1973). We show that DASL is sound and complete for full FOL. FOL requires infinite models in general, but we show that iterated finite sampling converges to correct truth values in the limit.

In this paper we show an application of DASL to learning from small amounts of data for two computer vision problems. The first problem is an illustrative toy problem based on the MNIST handwritten digit classification problem. The second is a well-known challenge problem of detecting visual relationships in images. In both cases, we demonstrate that the addition of declarative knowledge improves the performance of a vanilla DL model. This paper makes the following contributions:

1. The novel framework DASL, which compiles a network from declarative knowledge and bespoke domain-specific reusable component networks, enabling gradient-based learning of model components;

2. Grounding of the proposed framework in model theory, formally proving its soundness and completeness for full first order logic;

3. A logit representation of truth values that avoids vanishing gradients and allows deep logical structures for neural-symbolic systems;

4. Syntactic extensions that allow (i) restricted quantification over predicates and functions without violating first order logic constraints, and (ii) novel hybrid network architectures;

5. Evaluation on two computer vision problems with limited training data, demonstrating that knowledge reduces data requirements for learning deep models *e.g.* factor of 1000 for the MNIST toy problem and 10.7% improvement in accuracy for visual relationship detection in a data scarce setting.

2 RELATED WORK

Neuro-Symbolic Computing: Early efforts to augment DNNs with logic focused on propositional logic, which supports only logical connectives between (atomic) propositions Garcez et al. (2012; 2019). For example, KBANN Towell & Shavlik (1994) maps a set of propositions into a graph, constructs a neural network, and then trains it. DASL follows this basic idea but fully supports full first order logic (FOL) as well as arithmetic expressions.

Similar to several prior efforts Hu et al. (2016); Rocktäschel et al. (2015); Li & Srikumar (2019), DASL replaces Booleans with real-valued *pseudo-probabilities* to make the logical operations differentiable. This circumstance has motivated the invention of a collection of *ad hoc* aggregation operators for representing logical connectives Detyniecki (2001). These include the *t-norm*, used by Logic Tensor Networks (LTNs) Serafini & Garcez (2016) and the above works. Instead, DASL uses a logit representation for truth values, which avoids vanishing gradients more comprehensively than the logarithmic representations of Giannini et al. (2019); Rocktäschel et al. (2015), enabling learning in deeper logical structures. DASL also differs in supporting multiple entity types, arithmetic, and non-traditional operations such as *softmax* that enable richer interaction between the NN and knowledge (Section 4). DASL also represents the first time that soundness and completeness have been established for a FOL system applied to neural networks. There is also a body of work on tensor representations of logic over finite domains that is focused on efficient evaluation of formulas but is not concerned with learning Sato (2017); Nguyen et al. (2018).

Compositional DL: DASL is related to works that execute a task by composing trainable neural modules by parsing a query (in natural language) Andreas et al. (2016); Yi et al. (2018a); Mao et al. (2019); Yi et al. (2018b). For example, Yi et al. (2018b) focuses on visual question answering and employs a differentiable tree-structured logic representation, similar to DASL. However, it only does so to learn to translate questions, whereas DASL learns the semantics of the application domain and can also integrate useful domain knowledge.

Structured Learning: Other work exploits underlying structure in the data or the label space to learn DNNs using techniques such conditional random fields, graph neural networks, attention models, etc., including Belanger et al. (2017); Kim et al. (2017); Battaglia et al. (2018); Peng et al. (2018); Zheng et al. (2015). These methods impose structure by either adapting the DNN architecture Battaglia et al. (2018) or the loss function Zheng et al. (2015). DASL instead imposes soft constraints by compiling DNNs based on rules that can be stated in a flexible manner using FOL.

Semantic Reasoning: By the *semantics* of a logical language we mean an interpretation of its symbols (which do not include logical connectives and quantifiers); a *model* in the sense of model theory Weiss & D'Mello (1997). In common with several methods Xie et al. (2019), DASL grounds its entities in vector spaces (embeddings) and its predicates and functions in trainable modules. DASL builds on prior works on semantic representation techniques Pennington et al. (2014); Mikolov et al. (2013); Deerwester et al. (1990) by enabling logical statements to modify the entity embeddings so as to mirror semantic similarity in the application. Another research direction Rocktäschel & Riedel (2017); Cohen et al. (2017); de Jong & Sha (2019) is to structure DNNs and embedding spaces around proof trees to reduce the brittleness of theorem provers Siekmann & Wrightson (1983).

Bayesian Belief Networks: Substitution of pseudo-probabilities for Booleans fails to capture uncertainty the way fully Bayesian methods do Jaynes (2003). Bayesian Belief networks Pearl (2009) accurately represent probabilities but lack expressivity and face computability challenges. Bayes nets are most comfortably confined to propositional logic. Efforts to extend them to first-order logic include Markov Logic Networks Richardson & Domingos (2006), which use an undirected network to represent a distribution over a set of models, *i.e.*, *groundings* or *worlds* that can interpret a theory. The lifted inference approach Kimmig et al. (2004) reasons over populations of entities to render the grounded theory computationally tractable. These methods generally do not support the concept of (continuous) soft semantics through the use of semantic embedding spaces, as DASL does, although DeepProbLog Manhaeve et al. (2018) uses semantic embeddings and neural predicate representations within Bayesian Prolog rather than full FOL.

3 APPROACH

In this section we describe our approach to integrate data with relevant expert knowledge. Consider the task, depicted in Figure 1, of predicting the relationship between bounding boxes containing a subject and an object. In addition to learning from labeled training samples, we want to incorporate the commonsense knowledge that if the predicted relationship is "Riding" then the subject must be able to ride, the object must be ridable, and the subject must be above the object. Incorporating such knowledge results in a more robust model that uses high-level semantics to improve generalization and learn from a small number of examples. DASL achieves integration of the continuous representations in DNNs with the discrete representations typically used for knowledge representation by compiling a DNN from the knowledge assertions and grounding the vocabulary of the domain in component networks, enabling gradient-based learning of the model parameters. We begin by providing the theoretic underpinning of DASL in first order logic.

3.1 DASL MODEL THEORY

We follow a standard treatment of first order logic, starting with a language $\mathfrak{L}$ consisting of variables, constants, function symbols, and relation symbols. We use standard connectives $\neg$ ('not'), $\wedge$ ('and'), $\vee$ ('or'), $\rightarrow$ ('implies'), and quantifiers $\forall$ ('for all') and $\exists$ ('there exists'). For simplicity, we assume that no variable occurs bound twice, meaning that we would always need to rewrite $(\forall x)\varphi(x) \wedge (\exists x)\psi(x)$ as $(\forall x)\varphi(x) \wedge (\exists y)\psi(y)$. We treat $=$ ('equals') as a logical relation symbol. Terms are built up from variables, constants, and function symbols. Formulas are build up from terms, realtion symbols, and connectives. We restrict our attention to finite languages $\mathfrak{L}$ and only consider finite formulas over $\mathfrak{L}$.

Terms and formulas provide the syntax of first order logic. The semantics for classical logic is provided by boolean *Tarski models* [Chang & Keisler (1973), Weiss & D'Mello (1997)], a special case of infinite-valued Tarski models in which *truth values* range over the closed interval $\mathbb{T} = [0, 1]$ [Hájek (1998)]. A Tarski model $\mathfrak{A}$ with *domain* A consists of a function I which maps constants to elements of A, an n-ary function $I_f : A^n \rightarrow A$ for each n-ary function symbol f, and an n-ary function $I_R : A^n \rightarrow \mathbb{T}$ for each n-ary relation symbol R. We write $\mathfrak{A} = \langle A; I, I_{f_1}, \ldots, I_{f_m}, I_{R_1}, \ldots, I_{R_r}\rangle$ and $|\mathfrak{A}| = A$. A *sample function* maps variables to A; a model $\mathfrak{A}$ and set $\mathfrak{S}$ of sample functions together map terms to A and formulas to $\mathbb{T}$, written $(\mathfrak{A}, \mathfrak{S})[s]$ and $(\mathfrak{A}, \mathfrak{S})[\varphi]$.

The logical symbols need a fixed interpretation in order to specify a full semantics. We define $(\mathfrak{A}, \mathfrak{S})[\neg\varphi] = 1 - (\mathfrak{A}, \mathfrak{S})[\varphi]$; $(\mathfrak{A}, \mathfrak{S})[\varphi \wedge \psi] = (\mathfrak{A}, \mathfrak{S})[\varphi] * (\mathfrak{A}, \mathfrak{S})[\psi]$; and $(\mathfrak{A}, \mathfrak{S})[(\forall x)\varphi] = \prod_{u \in A}(\mathfrak{A}, \mathfrak{S} \restriction u/x)[\varphi]$ where $\mathfrak{S} \restriction u/x = \{\mathcal{S} \mid \mathcal{S} \in \mathfrak{S}, \mathcal{S}(x) = u\}$, so we are interpreting universal statements as (possibly infinite) conjunctions over all possible instantiations of the bound variable. $\rightarrow$, $\vee$, and $\exists$ are defined in terms of $\neg$, $\wedge$, and $\forall$. An interpretation of '$=$' is said to *preserve equality* if for any terms t and s, $(\mathfrak{A}, \mathfrak{S})[t = s] = 1$ if and only if $(\mathfrak{A}, \mathfrak{S})[t] = (\mathfrak{A}, \mathfrak{S})[s]$ for every $\mathfrak{A}$ and $\mathfrak{S}$. For now, we assume an equality-preserving interpretation, and we specify a particular interpretation below. We write $\mathfrak{S}^*(A)$ for the class of all sample functions over A and $\mathfrak{A}[\varphi]$ for $(\mathfrak{A}, \mathfrak{S}^*(|\mathfrak{A}|))[\varphi]$. $(\mathfrak{A}, \mathfrak{S})[\varphi]$ is well-defined whenever every $\mathcal{S} \in \mathfrak{S}$ is defined over every (bound and free) variable of φ and is consistent over every free variable of φ, and in particular $\mathfrak{A}[\varphi]$ is well-defined if and only if φ is *closed* (meaning that every variable of φ is bound by a quantifier).

Proposition 1 *Let φ be a closed formula. Every infinite-valued Tarski model $\mathfrak{A}$ such that $\mathfrak{A}[\varphi] = 1$ defines a classical boolean Tarski model of φ, and every boolean Tarski model of φ defines an infinite-valued Tarski model mapping φ to 1.*

This establishes that infinite-valued Tarski models fully capture the semantics of classical first order logic. While such models are not novel, their definitions vary significantly, and our use of sample functions instead of the usual variable interpretations is customized for our purposes below.

3.2 FINITARY COMPUTATION

DASL unifies machine learning and logical inference by generating infinite-valued Tarski models as described above which best fit both training data and logical formulas. A finite theory Γ is represented in DASL by the conjunction of its formulas, so we can focus on operation over a single formula φ. An embedding architecture is specified for the interpretation function I, and a network architecture is specified for each I_F. The arithmetic semantic operators described above are applied as specified by

the structure of φ, composing the interpretation networks into a single network, and this network is trained with a target output of 1.

We would like to prove that this technique completely captures full first order logic, but of course we know that some logical theories are only satisfied by infinite domains, and that all of our computation will be done over a finite number of finite samples. The constructed DASL network together with assigned parameter values specifies a unique model $\mathfrak{A}$, so any optimization technique that moves from one set of parameter assignments to another defines a sequence of models $\mathfrak{A}_0, \mathfrak{A}_1, \ldots$, each of which is evaluated over some finite set of samples, usually a "mini-batch" consisting of a relatively small subset of the domain. In order to relate this approach to infinitary models, we need to consider convergence in the limit. We begin by fixing a domain $A \subseteq \mathbb{R}^N$ for some N. If each $\mathfrak{A}_i$ is $\langle A; I_i, I_{1,i}, \ldots, I_{m,i}\rangle$ then we define $\lim_{i\to\infty} \mathfrak{A}_i = \langle A; \lim_{i\to\infty} I_i, \lim_{i\to\infty} I_{1,i}, \ldots, \lim_{i\to\infty} I_{m,i}\rangle$.

Proposition 2 (Soundness) *Let $\mathfrak{A}_0, \ldots$ be a model sequence such that $\lim_{i\to\infty} \mathfrak{A}_i = \mathfrak{A}^*$ and $\mathfrak{S} = \bigcup_{i=0}^{\infty} \mathfrak{S}_i$ be a sample set such that every $\mathcal{S} \in \mathfrak{S}$ occurs in infinitely many of the $\mathfrak{S}_i$. For any formula φ, if $\lim_{i\to\infty}(\mathfrak{A}_i, \mathfrak{S}_i)[\varphi] = 1$ then $(\mathfrak{A}^*, \mathfrak{S})[\varphi] = 1$.*

The above proposition assumes that all possible samples are continually revisited, but makes no other assumptions about sampling. It makes no assumptions about how we train models or whether training converges. Together with proposition 1, this shows that when DASL models do converge on finite sampling, they converge to a boolean Tarski model.

Proposition 3 (Completeness) *For any boolean model $\mathfrak{A}$ and formula φ such that $\mathfrak{A}[\varphi] = 1$, there is a family of finite sample sets $\mathfrak{S}_0, \ldots$ such that $\bigcup_{i=0}^{\infty} \mathfrak{S}_i = \mathfrak{S}^*(|\mathfrak{A}|)$ and $(\mathfrak{A}, \mathfrak{S}_i)[\varphi] = 1$ for all i.*

The above proposition is proved by including in each set the sample function which contains a witness for each existentially bound variable whenever one exists (see supplementary section A.1.1 for details). This tells us that for any satisfiable formula, we can find the model that satisfies it even with only finite sampling.

3.3 Implementing DASL Networks

We implement DASL in the popular deep learning library PyTorch Paszke et al. (2019). Any model that can be implemented in PyTorch can be used for the interpretation functions. We use a PyTorch `dataloader` to generate the mini-batches $\mathfrak{S}_i$ on which eachmodel $\mathfrak{A}_i$ is evaluated.

Loss function. A simple and intuitive loss would be $\mathcal{L}(\mathfrak{A}, \mathfrak{S}, \varphi) = 1 - (\mathfrak{A}, \mathfrak{S})[\varphi]$. This loss works as desired at 0 and 1, but if $(\mathfrak{A}, \mathcal{S})[\varphi] = t$ for every $\mathcal{S} \in \mathfrak{S}$ then $(\mathfrak{A}, \mathfrak{S})[(\forall x)\varphi] = t^{|\mathfrak{S}|}$ which has the disadvantage that loss depends on the sample size and that it gets close to 1 as the sample size increases which can cause rounding errors to force a 0 gradient. Both of these challenges are addressed by using a logit representation of the truth value and using cross-entropy as a loss function. For $t \in \mathbb{T}$ define $\mathrm{logit}(t) = \ln \frac{t}{1-t}$ and $\sigma(x) = \frac{1}{1+e^{-x}}$. Then $\sigma(\mathrm{logit}(t)) = t$ and $\mathrm{logit}(\sigma(x)) = x$. The cross entropy between the distributions $(t, 1-t)$ and $(1, 0)$ is $-\ln(t)$, so when $t = (\mathfrak{A}, \mathfrak{S})[\varphi]$ we define $\mathcal{L}(\mathfrak{A}, \mathfrak{S}, \varphi) = -\ln(t) = \ln(1 + e^{-\mathrm{logit}(t)})$. We reason directly in logit space when computing the loss, deriving formulas from the above definitions (see supplementary section A.2 for details).

Equality. DASL functions can include standard regression tasks $f(x) = y$ for real-valued y. This requires a differentiable interpretation that preserves equality. For (scalar-valued) terms u and v, we model $d = (\mathfrak{A}, \mathfrak{S})[u] - (\mathfrak{A}, \mathfrak{S})[v]$ by a normal distribution with mean 0 and variance ε_1^2 when equality holds and by a normal distribution with mean μ and variance ε_2^2 otherwise. We define the logit as the log ratio of the resulting densities for $x = |d|$: $\mathrm{logit}((\mathfrak{A}, \mathfrak{S})[u = v]) = \ln \frac{2\varepsilon_2^2}{\varepsilon_1} + \frac{x^2}{2\varepsilon_1} - \ln(e^{-(x-\mu)^2/2\varepsilon_2^2} + e^{-(x+\mu)^2/2\varepsilon_2^2})$. For vector-valued u and v, we use the same distribution with $x = ||u - v||$. This interpretation approaches preserving equality as $\varepsilon_1 \to 0$.

Extending the logical language. We implement many-sorted logic for convenience and efficiency. Sorts are subsets of the domain. Each quantifier ranges over a specified subset. Quantifiers can map variables to elements of tuples from enumerated sets, so a standard machine learning problem can be written as $(\forall(x, y) : D)f(x) = y$, where D lists the (x, y) training pairs.

In addition to learned models, arbitrary deterministic functions can be coded in PyTorch. This allows us to fix logical interpretations of $<, >, +, -, \times$, etc. We allow propositional constants (formulas

of the language which are always interpreted either as 0 or as 1) and we allow $\wedge$ and $\vee$ to apply to arbitrary sequences of formulas. The connectives also generalize to work component-wise on tensors and support broadcasting as is familiar for tensor operators in PyTorch. For example, if $\boldsymbol{X}$ is a matrix and $\boldsymbol{y}$ is a vector, both of logits, then $\boldsymbol{X} \wedge \boldsymbol{y} = \boldsymbol{Z}$, where $Z_{ij} = X_{ij} \wedge y_i$.

Finally, we provide an explicit operator $softselect(\Gamma, i)$ (denoted as $\pi_i(\Gamma)$) which outputs the logit value for the ith formula of Γ after application of the logit version of the softmax operator. This allows us to directly specify standard architectures for multi-class classification problems and to allow rules to operate on the classifier output within the model. Because i is an integer argument, we can quantify over it, effectively quantifying over a fixed finite list of predicates, providing syntactic convenience without violating the constraints of first order logic.

4 EXPERIMENTS

We evaluate DASL on two computer vision problems in data scarce conditions. The first task is a toy problem based on MNIST digit classification, where knowledge is provided as an arithmetic relation satisfied by unlabeled triplets of digit images that are arranged to satisfy that relation (subsection 4.1). We then focus on the problem of detecting visual relationships between object pairs and use commonsense knowledge about the plausible arguments of the relationship (subsection 4.2).

4.1 TOY EXAMPLE ON MNIST

Problem statement: We use a toy example to demonstrate DASL's ability to train a NN from a few labeled samples and large number of unlabeled samples satisfying a rule. We denote a grayscale input image of a MNIST digit LeCun et al. (1998) as $\boldsymbol{X}$ and its label (if provided) as $y(\boldsymbol{X}) \in \mathbb{Z}_{10}$, where $\mathbb{Z}_{10} = \{0, 1, ..., 9\}$. The task is to learn a NN $\text{digit}(\boldsymbol{X})$ to predict the digit in a test image.

We split the training data ($50K$ images) into two disjoint sets: $\textbf{Labeled}$, containing a small number N_{tr} of labeled examples per digit class, and $\textbf{Unlabeled}$, used to generate the set $\textbf{Triples}$ containing triplets of images $(\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3)$ satisfying the rule $y(\boldsymbol{X}_1) + y(\boldsymbol{X}_2) = y(\boldsymbol{X}_3) \bmod 10$. $\textbf{Triples}$ contains only unlabeled images that together satisfy this relationship. We wish to learn the classifier by using $\textbf{Labeled}$ and $\textbf{Triples}$, and thus the challenge is to compensate for the small size of $\textbf{Labeled}$ by leveraging the prior knowledge about how the unlabeled images in $\textbf{Triples}$ are related. We formulate this problem within DASL by using its $softselect$ operator π_i that, applied to the NN output $\text{digit}(\boldsymbol{X})$, returns the normalized score for the ith class. This rule is written as:

$$
\begin{aligned}
&(\forall(\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3) : \textbf{Triples})(\forall y_1 : \mathbb{Z}_{10})(\forall y_2 : \mathbb{Z}_{10}) \\
&\quad [(\pi_{y_1}(\text{digit}(\boldsymbol{X}_1)) \wedge \pi_{y_2}(\text{digit}(\boldsymbol{X}_2))) \\
&\qquad \rightarrow \pi_{(y_1 + y_2) \bmod 10}(\text{digit}(\boldsymbol{X}_3))]
\end{aligned}
$$

We quantify over the triplets from $\textbf{Triples}$ and all possible pairs of digits from $\mathbb{Z}_{10}$. We use this theory to augment the theory corresponding to the labeled training examples $(\forall(\boldsymbol{X}) : \textbf{Labeled})(\pi_{y(\boldsymbol{X})}(\text{digit}(\boldsymbol{X})))$. The model is required to correctly infer the (unknown) labels of the triplet members and then use them for indirect supervision. We evaluate the model using the average accuracy on the test set ($10K$ images). For $\text{digit}(\boldsymbol{X})$, we used a two-layer perceptron with 512 hidden units and a sigmoid non-linearity. We performed experiments in data scarce settings with $N_{tr} = 2, 5, 10$, and 20, and report mean performance with standard deviation across 5 random training subsets as shown in Figure 2. We use an equal number of examples per-class for constructing the triplets. We use a curriculum based training strategy to prevent the model from collapsing to a degenerate solution, especially for lower values of N_{tr} (see supplementary section A.4 for details). We report performance after $30K$ training iterations. A test image is classified into the maximum scoring class.

Results: Figure 2 shows a plot of digit classification accuracy versus the number of samples per class used for creating the triplets. We observe that the NN trained with both knowledge and data (*With-knowledge*) outperforms its counterpart trained with only labeled samples (*No-knowledge*). The improvement is particularly notable when training with smaller labeled training sets; *e.g.*, for $N_{tr} = 2$, using all the knowledge raises performance from 53.3 ± 1.01 to 97.7 ± 0.00. We also note that the performance of the *With-knowledge* model improves as the number of triplets increases

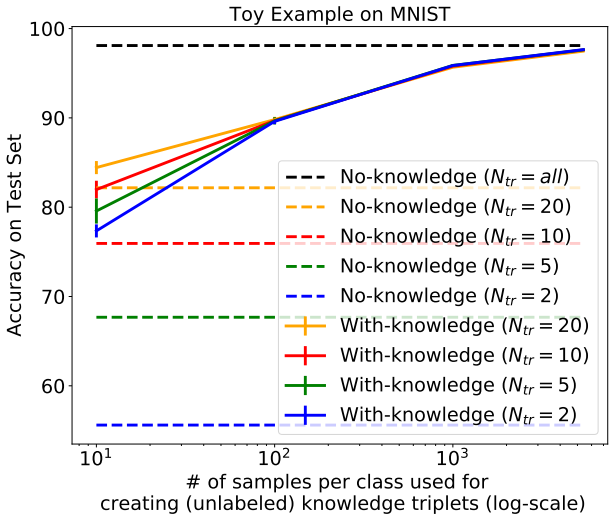

Figure 2: Figure showing the results for the MNIST toy example with a plot of accuracy of digit classification versus number of samples per class used for creating the unlabeled knowledge triplets. The labels *With-knowledge* and *No-knowledge* denote whether the training included the knowledge-augmented unlabeled triplets satisfying the given modular arithmetic (see subsection 4.1). N_{tr} refers to the number of labeled training examples per class (*all* refers to the entire training set). Best seen in color.

and converges to similar values for different values of N_{tr}, indicating that the knowledge renders extra labels largely superfluous. The mean performance is 97.6 ± 0.00, which is competitive with the performance of a model trained with all $50K$ labeled examples in MNIST (98.1 for $N_{tr} = all$). Results on a related problem were also shown in Manhaeve et al. (2018).

4.2 Visual Relationship Detection

Problem Statement: We use the Visual Relationship Detection (VRD) benchmark Lu et al. (2016) to evaluate the **Predicate Detection Task:** Given an image and a set of bounding boxes with object category labels, predict the predicates that describe the relationships within object pairs. The VRD dataset contains 5000 images spanning 37993 relationships covering 100 object classes and 70 predicate classes. We use splits provided by the authors that contain 4000 train and 1000 test images. The dataset also provides a *zero-shot* test subset of 1877 relationships built from the same classes as the training data but containing novel combinations of predicate classes with object class pairs.

Baseline model: We begin with a NN $\mathrm{vrd}(\boldsymbol{I}, s, o)$ that outputs raw scores for predicate classes, where $\boldsymbol{I}$ is the input RGB image and s and o are the indices of the subject and object classes respectively. We implement two variants of vrd similar to that proposed in Liang et al. (2018). The first variant, referred to as *VGG*, extracts visual features from the last layer of a pre-trained VGG-16 network from the bounding box of the subject, the object, and their union. These features are projected into a 256 dimensional space by using a projection layer P (made of a fully-connected (FC) layer and a ReLU non-linearity) and then fused by concatenation. The features are passed through another P layer followed by a FC layer to predict the class-wise scores. The second variant, referred to as *VGG-SS*, additionally incorporates the word-embedding features of the subject and the object (300 dimensional Glove features Pennington et al. (2014)) along with the normalized relative spatial coordinates (see supplementary A.5 for details). These features are first projected using additional P layers and then concatenated with visual features, as done for VGG, prior to predicting the class-scores.

We express knowledge as vectors of formulas as discussed in subsection 3.3. We begin by defining $\mathrm{CanRide}$ as a constant vector of truth values for all objects which is *True* at indices of objects which can ride and *False* elsewhere. $\mathrm{CanRide}(s)$ selects its sth element. Similarly, we define $\mathrm{Ridable}$ as a vector which is *True* at exactly the indices of objects that can be ridden. Finally we define a one-hot vector of truth values $\boldsymbol{h}_{Cls} \in \mathbb{R}^{70}$, which is *True* at the index of the predicate class "Cls" and *False* elsewhere. The theory that asserts that vrd should output the class labels assigned in the training data

and that the "Riding" predicate should only apply when the subject can ride and the object can be ridden is written as:

$$(\forall(\boldsymbol{I}, s, o, y) : \mathbf{D}_{\mathrm{vrd}})[\pi_y(\mathrm{vrd}(\boldsymbol{I}, s, o) \\ \wedge (\boldsymbol{h}_{\mathrm{Riding}} \rightarrow \mathrm{CanRide}(s) \\ \wedge \mathrm{Ridable}(o)))]$$

where y is the given training label and $\mathbf{D}_{\mathrm{vrd}}$ is the training dataset. This rule reduces the learning burden of the classifier for "Riding" class by allowing feedback only when $\mathrm{CanRide}(s)$ is *True*. We introduce a few more rules by adding them in conjunction with the above rules (see supplementary).

Evaluation: We follow Yu et al. (2017), reporting Recall@N (R@N), the recall of the top-N prediction scores in an image where we take into account all 70 predictions per object pair. This strategy is different from Lu et al. (2016), which only considers the top prediction for each object pair penalizing cases where multiple predicates apply equally well but were omitted by the annotators.

Method	R@50		R@100	
	Standard	Zero-Shot	Standard	Zero-Shot
	1% Data			
VGG (baseline)	60.8 ± 6.7	40.7 ± 5.8	75.4 ± 7.8	59.4 ± 8.1
+ Knowledge	$68.5 \pm 1.8^{**}$	$49.5 \pm 1.5^{**}$	$83.1 \pm 1.6^{**}$	$70.1 \pm 2.4^{**}$
VGG-SS (baseline)	67.9 ± 8.5	47.6 ± 8.5	80.3 ± 7.6	65.6 ± 9.2
+ Knowledge	$74.0 \pm 0.7^{*}$	$54.4 \pm 1.4^{*}$	$85.9 \pm 0.5^{*}$	$73.4 \pm 1.2^{*}$
	5% Data			
VGG (baseline)	70.3 ± 0.5	48.4 ± 1.0	83.5 ± 0.4	68.3 ± 0.9
+ Knowledge	$73.8 \pm 0.5^{**}$	$53.4 \pm 0.9^{**}$	$86.4 \pm 0.4^{**}$	$73.7 \pm 1.1^{**}$
VGG-SS (baseline)	79.6 ± 0.4	58.1 ± 1.2	89.6 ± 0.3	77.1 ± 1.1
+ Knowledge	79.9 ± 0.4	$59.6 \pm 0.9^{**}$	89.7 ± 0.3	$78.5 \pm 0.8^{**}$

Table 1: Performance on the predicate detection task Lu et al. (2016). We report Recall@N averaged (with standard deviation) across 10 random subsets. We report the statistical significance between "baseline" and corresponding knowledge augmented model ("+ Knowledge") (p-value < 0.01 as $**$ and p-value < 0.05 as $*$).

Results: Table 1 shows the results on the VRD dataset when training with knowledge (+ *Knowledge*) and without knowledge (*baseline*). We observe consistent improvements across all cases with augmentation of knowledge. The improvements are higher for the 1% data ($+7.7\%$ for R@100 for Standard) than the 5% data ($+2.9\%$ for R@100 for Standard) showing that knowledge has more benefits in lower data regimes. We made similar observation for the MNIST toy example. The improvements are generally higher for the zero-shot setting ($+10.7\%$ for R@100 in the 1% case) since this setting is inherently data starved. We also note that the improvements are comparatively smaller for the VGG-SS network since semantic and spatial information are being explicitly injected as features into the model.

5 Conclusion

We introduced **Deep Adaptive Semantic Logic (DASL)** to unify machine reasoning and machine learning. DASL is fully general, encompassing all of first order logic and arbitrary deep learning architectures. It improves deep learning by supplementing training data with declarative knowledge expressed in first order logic. The vocabulary of the domain is realized as a collection of neural networks. DASL composes these networks into a single DNN and applies backpropagation to satisfy both data and knowledge. We provided a formal grounding that demonstrates the correctness and full generality of DASL for the representation of declarative knowledge in first order logic, including correctness of mini-batch sampling for arbitrary domains. This gives us the freedom to apply DASL to new domains without requiring new correctness analysis.

We demonstrated a 1000-fold reduction in data requirements on the MNIST digit classification task by using declarative knowledge in the form of arithmetic relations satisfied by unlabeled image triplets. We then demonstrated the application of commonsense knowledge to visual relationship detection, improving recall from 59.4% to 70.1%. Here, the commonsense knowledge enabled us to better focus the model on cases not covered by knowledge.

REFERENCES

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *CVPR*, pp. 39–48. IEEE Computer Society, 2016. ISBN 978-1-4673-8851-1.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

David Belanger, Bishan Yang, and Andrew McCallum. End-to-end learning for structured prediction energy networks. In *ICML*, pp. 429–439. JMLR. org, 2017.

Chen Chung Chang and H Jerome Keisler. *Model theory*. Elsevier, 1973.

William W Cohen, Fan Yang, and Kathryn Rivard Mazaitis. Tensorlog: Deep learning meets probabilistic dbs. *arXiv preprint arXiv:1707.05390*, 2017.

Michiel de Jong and Fei Sha. Neural theorem provers do not learn rules without exploration. *arXiv preprint arXiv:1906.06805*, 2019.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of American Society for Information Science*, 41:391–407, 1990.

Marcin Detyniecki. Fundamentals on aggregation operators. Technical report, University of California, Berkeley, 2001.

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

Artur S. d'Avila Garcez, Marco Gori, Luís C. Lamb, Luciano Serafini, Michael Spranger, and Son N. Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *Human Level Artificial Intelligence*, 1:1–10, 2019.

M. R. Genesereth and N. J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, 1987.

Francesco Giannini, Giuseppe Marra, Michelangelo Diligenti, Marco Maggini, and Marco Gori. On the relation between loss functions and t-norms. *arXiv preprint arXiv:1907.07904*, 2019.

P. Hájek. *Metamathematics of Fuzzy Logic*. Trends in Logic. Springer Netherlands, 1998. ISBN 9781402003707.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. *arXiv preprint arXiv:1603.06318*, 2016.

E. T. Jaynes. *Probability theory: The logic of science*. Cambridge University Press, Cambridge, 2003.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

Angelika Kimmig, Lilyana Mihalkova, and Lise Getoor. Lifted graphical models: A survey. *Machine Learning*, 99:1–45, 2004.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

Tao Li and Vivek Srikumar. Augmenting neural networks with first-order logic. *arXiv preprint arXiv:1906.06298*, 2019.

Kongming Liang, Yuhong Guo, Hong Chang, and Xilin Chen. Visual relationship detection with deep structural ranking. In *AAAI*, 2018.

Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, pp. 852–869. Springer, 2016.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimming, Tomas Demeester, and Luc DeRaedt. Deepproblog: Neural probabilistic logic programming. In *NeurIPS 2018*, 2018.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *ICLR*. OpenReview.net, 2019.

Gary Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *NIPS*, pp. 3111–3119. Curran Associates, Inc., 2013.

Hien D. Nguyen, Chiaki Sakama, Taisuke Sato, and Katsumi Inoue. Computing logic programming semantics in linear algebra. In *Lecture Notes in Artificial Intelligence 11248*. 2th International Conference on Multi-disciplinary Trends in Artificial Intelligence (MIWAI 2018), Springer, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*, pp. 8024–8035, 2019.

Judea Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann, San Francisco, Calif., 2009. ISBN 9781558604797 1558604790.

Hao Peng, Sam Thomson, and Noah A Smith. Backpropagating through structured argmax using a spigot. *arXiv preprint arXiv:1805.04658*, 2018.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, pp. 1532–1543, 2014.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2): 107–136, 2006. ISSN 0885-6125. doi: 10.1007/s10994-006-5833-1.

Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *NIPS*, pp. 3788–3800. Curran Associates, Inc., 2017.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *ICML*, pp. 1119–1129, 2015.

Taisuke Sato. Embedding tarskian semantics in vector spaces. In *AAAI-17 Workshop on Symbolic Inference and Optimization*, 2017.

Luciano Serafini and Artur S. d'Avila Garcez. Logic tensor networks: Deep learning and logical reasoning from data and knowledge. *CoRR*, abs/1606.04422, 2016.

Jörg Siekmann and G. Wrightson. *Automation of Reasoning*. Springer, 1983.

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

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

William Weiss and Cherie D'Mello. Fundamentals of model theory, 1997.

Yaqi Xie, Ziwei Xu, Kuldeep Meel, Mohan S. Kankanhalli, and Harold Soh. Semantically-regularized logic graph embeddings. *CoRR*, abs/1909.01161, 2019.

Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NIPS*, pp. 2319–2328, 2017.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *NIPS*, pp. 1039–1050, 2018a.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *NIPS*, pp. 1031–1042, 2018b.

Ruichi Yu, Ang Li, Vlad I Morariu, and Larry S Davis. Visual relationship detection with internal and external linguistic knowledge distillation. In *CVPR*, pp. 1974–1982, 2017.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *ICCV*, pp. 1529–1537, 2015.