




# Symbolic Deep Networks: A Psychologically Inspired Lightweight and Efficient Approach to Deep Learning

Vladislav D. Veksler,<sup>a,b</sup>  Blaine E. Hoffman,<sup>b</sup> Norbou Buchler<sup>b</sup>

<sup>a</sup>*DCS Corp, Alexandria, VA*

<sup>b</sup>*Human Systems Integration Division (HSID), U.S. Army DEVCOM Data & Analysis Center (DAC)*

Received 17 December 2020; received in revised form 6 August 2021; accepted 9 August 2021

---

## Abstract

The last two decades have produced unprecedented successes in the fields of artificial intelligence and machine learning (ML), due almost entirely to advances in deep neural networks (DNNs). Deep hierarchical memory networks are not a novel concept in cognitive science and can be traced back more than a half century to Simon's early work on discrimination nets for simulating human expertise. The major difference between DNNs and the deep memory nets meant for explaining human cognition is that the latter are symbolic networks meant to model the dynamics of human memory and learning. Cognition-inspired symbolic deep networks (SDNs) address several known issues with DNNs, including (1) learning efficiency, where a much larger number of training examples are required for DNNs than would be expected for a human; (2) catastrophic interference, where what is learned by a DNN gets unlearned when a new problem is presented; and (3) explainability, where there is no way to explain what is learned by a DNN. This paper explores whether SDNs can achieve similar classification accuracy performance to DNNs across several popular ML datasets and discusses the strengths and weaknesses of each approach. Simulations reveal that (1) SDNs provide similar accuracy to DNNs in most cases, (2) SDNs are far more efficient than DNNs, (3) SDNs are as robust as DNNs to irrelevant/noisy attributes in the data, and (4) SDNs are far more robust to catastrophic interference than DNNs. We conclude that SDNs offer a promising path toward human-level accuracy and efficiency in category learning. More generally, ML frameworks could stand to benefit from cognitively inspired approaches, borrowing more features and functionality from models meant to simulate and explain human learning.

**Keywords:** Artificial Intelligence; Categorization; Cognitive Architectures; Deep Learning; Machine Learning; Supervised Learning; Symbolic Deep Learning; XAI

---

## 1. Introduction

The purpose of this paper is to introduce the cognitively inspired symbolic analog of sub-symbolic deep neural networks (DNNs or deep learning) as a framework useful for the types of learning/classification problems that commonly fall under the domain of machine learning (ML). Deep/hierarchical networks of linked symbolic information are not a new concept in artificial intelligence (AI), and go back as far as Simon's original work on discrimination nets at the very infancy of the field in the middle of the twentieth century (Chase & Simon, 1973; Feigenbaum & Simon, 1984; Simon, 1967; Simon & Feigenbaum, 1964; Simon & Gilmartin, 1973). During the AI winter of the 1980s, symbolic approaches became largely geared toward hand-coded expert systems that were limited in scope and scalability, and in the following decades, statistical sub-symbolic ML emerged as the more general and scalable path forward.

DNNs, in particular, have garnered much attention over the last decade due to the ability to detect signal among large, noisy input vectors and to classify complex input feature interactions (Rusk, 2015). Among much other success in data analysis and image/speech recognition, the ability of DNN to learn and play 49 different Atari games based solely on raw-pixel input (Mnih et al., 2015) and to learn the game of Go from scratch and beat a human world champion of Go (Byford, 2016) was transformational for the fields of artificial intelligence and cognitive science. DNN dominance was further solidified with a steady increase in availability of high-level libraries that run on personal computers (e.g., Abadi et al., 2016). As the field currently stands, articles and presentations having to do with DNNs dominate a vast majority of AI-related conferences.

Concurrently, symbolic memory/learning approaches<sup>1</sup> have gained dominance in computational cognitive modeling—a field that falls under the psychology domain to a greater degree than that of computer science. In part, this is because, unlike sub-symbolic neural nets, symbolic memory frameworks are explainable, enabling researchers to peek inside the model to understand why it does what it does. In part, this is simply due to the continuing success of using such frameworks to simulate human data across a wide range of cognitive and behavioral phenomena, including learning and memory, perception and attention, problem solving and decision making, language processing, etc. (e.g., Anderson, 2007). As the field currently stands, articles and presentations having to do with symbolic cognitive frameworks like ACT-R (Anderson, 2007) and SOAR (Laird, 2012) dominate cognitive modeling-related conferences.

Poor learning efficiency remains an Achilles' heel for DNNs and a problem that the ML community would very much like to solve. DNNs cannot learn from just a few or even a few dozen examples the way that humans can. One solution that is often employed in DNN research to aid in domains where few learning examples are available is to increase the number of epochs (i.e., train on the same few examples lots of times), though depending on the dataset, this solution often leads to overfitting (i.e., inability to generalize and accurately handle unseen data). Moreover, increasing the number of epochs increases the amount of time required to train a DNN, such that even for a small dataset, training times become prohibitive for dynamic applications. Ironically, symbolic approaches, though largely ignored in the ML

community, are a popular methodology for behavioral simulations in large part due to their ability to simulate the efficiency of human/animal learning.

Two other major problems for DNNs that are core strengths of symbolic approaches are the lack of explainability and a potential for catastrophic interference.<sup>2</sup> The explainability of what an agent holds in memory, what it's learning, and why it's making its decisions is relatively accessible in symbolic memory systems. Additionally, symbolic memory systems can add new memory nodes and links without any effect on the rest of the memory network, making this approach more robust to catastrophic interference than DNNs.

On the other hand, symbolic approaches are not typically applied to problems involving large, noisy analog input vectors. This greatly limits the domain types where symbolic approaches are employed.

Symbolic deep network (SDNs) is an approach that lies at the intersection of ML and cognitive modeling. SDNs combine the benefits of the hierarchical multi-layer (deep) structures found in the DNN frameworks with the learning efficiency and explainability of symbolic cognitive frameworks. SDNs are a more efficient, lighter, faster, and more explainable alternative to DNNs. The rest of this paper describes the SDN approach and contrasts it with DNN in terms of theory, accuracy, efficiency, robustness to noise, and catastrophic interference.

## 2. Symbolic deep learning

A symbolic version of deep learning is promising in that this method is capable of building classifiers from a small number of examples (d'Avila Garcez, Dutra, & Alonso, 2018; Dutra, Garcez, & D'Avila Garcez, 2017; Veksler & Buchler, 2019; Veksler et al., 2020; Zhang & Sornette, 2017). In this way, SDN learning efficiency is much closer than DNN to that of humans. Moreover, SDN memory is incremental (i.e., does not require a pre-specified size of the network) and is thus robust against catastrophic interference. Finally, symbolic memory lends itself to human interpretation, thus addressing issues relating to eXplainable artificial intelligence (XAI). Essentially, SDN addresses all of the traditional DNN limitations and is a promising avenue for ML research.

More specifically, a traditional DNN has a pre-specified number of layers, with a pre-specified number of nodes in each layer, and with nodes of each pair of successive layers being fully interconnected via weighted links (see Figure 1, left). As the network learns, the weights on these links change, but at no point can one look at those links and comprehend what exactly the network has learned. Because all knowledge is distributed among the links, the network has to be large enough to be able to learn a given problem and thus requires thousands or millions of iterations to learn even simple input–output mapping. The fully distributed nature of DNN memory also means that DNNs are prone to catastrophic interference and that a DNN's interior layers are a black box.

Symbolic deep nets, on the other hand, start with no nodes between input and output layers and learn these nodes based on perceived input node co-occurrences (see Figure 1, middle and right). These deep nodes are essentially combinations of input features (a.k.a., chunks or configural cues), and in the case of modeling human memory, deeper chunks are taken to

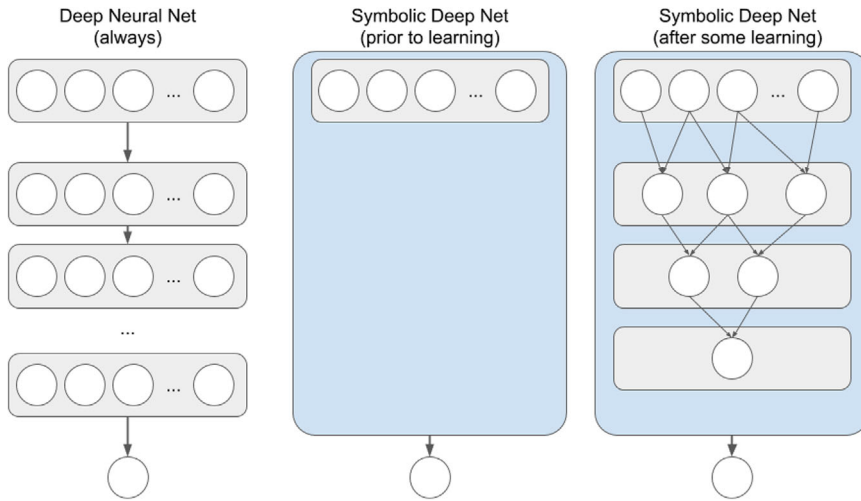


Fig. 1. Traditional deep neural net and symbolic deep net structures for networks with a single output node. Each circle represents a network node. Top row of each net represents the input layer, and bottom row is output nodes. Thicker arrows going from a node container to a node or another container represent a fully interconnected vector/matrix of weighted feed-forward links. Thinner arrows between nodes represent symbolic (i.e., not weighted) feed-forward links.

represent deeper domain expertise (Feigenbaum & Simon, 1984; Gobet, 1998). Due to the symbolic nature of chunk-based hierarchical memory, one can look at the learned chunks at any time so as to gain insight into what the network has learned. Because of the nature of chunk learning (one-shot learning), simple feature combinations can be learned quickly, enabling symbolic nets to learn at speeds on par with human learning—from just a few examples, rather than tens of thousands. Additionally, since knowledge in SDNs is only partially rather than fully distributed, SDNs are not as prone to catastrophic interference as DNNs.

Symbolic hierarchical representations have a long history in cognitive science literature. Some of these were integrated as models of memory without action-selection (e.g., Feigenbaum & Simon, 1984; Gobet & Lane, 2005). Such purely declarative models are more useful for predicting recognition than the type of input–output mapping that DNNs excel at.

Integrated cognitive architectures that include both state recognition and action-selection often include hierarchical memory systems as well. For example, declarative memory chunks in the ACT-R cognitive architecture are symbolic memory elements that are, in fact, sets of links to lower level chunks (Anderson, 1993; Anderson & Lebiere, 1998). Unfortunately, cognitive architectures were designed as systems for hand-coding of memory models rather than for solving problems out-of-the-box automatically (e.g., Anderson, 2007; Laird, 2012).

The most promising models of hierarchical learning/memory systems for the purposes of SDN system development and automatic model generation may be found in the categorization research domain. Models in the categorization literature were specifically developed with the purpose of mapping multi-feature inputs onto decisions (e.g., Gluck & Bower, 1988; Nosofsky, Gluck, Palmeri, McKinley, & Glauthier, 1994).

The greatest problem facing such hierarchical symbolic memory systems is one of computational limitations. For example, the configural-cue model of memory (Gluck & Bower, 1988) creates a configural node (i.e., chunk) for every unique set of potential inputs, thus creating a maximum of  $(k + 1)^n - 1$  memory chunks, where  $n$  is the number of input dimensions and  $k$  is the number of possible input values along each input dimension.<sup>3</sup> Although this exponential memory growth is concerning for large-input domains (e.g., image recognition), it may not be a problem for classification tasks with smaller input vectors, especially given modern computing power.

Perhaps, more important than the raw computational power available today, efficiency can be gained in SDNs by creating memory chunks only when they prove necessary. For example, Veksler, Gluck, Myers, Harris, and Mielke (2014) proposed a conservative-rational incremental memory system that reduces memory size, especially in noisy environments. Such memory reduction is exponential, improving efficiency by several factors of magnitude and greatly reducing the concern over computational limitations for SDNs.

Specifically, the conservative-rational model was proposed as an amalgamation of the configural-cue memory structure (Gluck & Bower, 1988), the ACT-R cognitive architecture chunk activation mechanism (Anderson & Lebiere, 1998), and reinforcement learning (RL) output utility learning (Sutton & Barto, 1998). The model starts with no deep nodes, only input and output layers. Each node in the input layer is a symbolic input, not numeric—either it is present or absent (numeric values in input vectors have to be binned and tiled so as to be translated into symbolic inputs). New deep nodes (i.e., chunks) are created over time as children of other inputs and chunks. To be exact, after feed-forward activation (i.e., perception) a chunk is created and linked from two of the deepest nodes with the highest activation, where activation is calculated based on each node's perceived frequency and recency (e.g., the model may be exposed to the inputs *large*, *square*, and *white* a few times, learn the chunk  $\{large, square\}$  and add it to the second layer, and then add the chunk  $\{large, square, white\}$  to the third layer, such that future feed-forward activation will spread down the links from *large*, *square*, and *white* inputs to activate both  $\{large, square\}$  and  $\{large, square, white\}$  chunks).

To provide a concrete example of how this deep memory structure works in combination with RL for choosing among outputs, and also to give an example of SDN explainability, let us imagine that the model is exposed to lots of *white*, *large*, and *square* inputs (among others), and in choosing between outputs *A* and *B* sometimes it received a reward, and sometimes it did not. However, whenever *white* and *large* inputs co-occurred and the model chose *A*, more often than not the model received a reward. Then the model is presented with a white large square, activating inputs/chunks  $[large, square, white, \{large, square\}, \{large, square, white\}]$ , and selecting *A* as the output. At this point, we may be able to observe the entire snapshot of active memory, as shown in Figure 2, and determine at a glance that the  $\{large, square\} \rightarrow A$  chunk-action utility was the determining factor in choosing *A* as the output.

All inputs, chunks, and links to chunks in the conservative-rational model are symbolic, not numeric. All links to the output nodes, whether from input layer nodes or from deeper chunk nodes, are numeric weights signifying RL state-action utilities. After the model makes

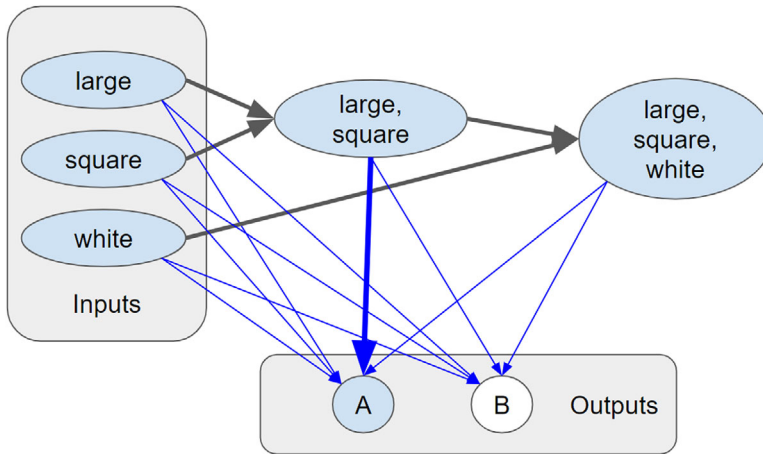


Fig. 2. Sample snapshot of active memory nodes in the conservative-rational model. Symbolic links between inputs and chunks shown in black. RL chunk-action utilities shown as weighted connections in blue. In this example, the model is presented with inputs *white*, *large*, *square*, and in choosing between outputs *A* and *B*, output *A* had the higher average utility, with the  $\{large, square\} \rightarrow A$  chunk-action utility being the major determining factor.

a decision, the output links are updated via error-driven learning (see Veksler et al., 2014, for a more detailed description of RL in the conservative-rational model). There is no calculation performed during feed-forward activation in this model, either on the links or in the nodes. There is no back-propagation in this model either, just reward learning after response feedback. Thus, this is a much more lightweight approach to deep learning, greatly reducing memory and computation time for both training and predictions.

For the purposes of examining the usefulness of psychologically inspired SDNs in the ML domain, we will employ the conservative-rational model as our SDN of choice. In part, this is because this model seems to embody the idea of SDN in that it is a purely symbolic deep memory network (sans the RL layer), and in part because it is an amalgamation of several other memory models from psychological literature. Additionally, prior comparisons of DNN with the conservative-rational model (Veksler et al., 2020) have suggested that SDN provides efficiency benefits over DNN and have served as the inspiration for the more in-depth evaluation pursued in the simulations below. We presume that performance on all datasets where we test SDN in this paper can be improved with tweaks to the algorithm, but the purpose here is not to find the optimal SDN configuration and performance, but rather to show that a purely symbolic deep structure can be employed to achieve excellent results on various ML problems, and do so more efficiently than DNNs.

### 3. Simulations

For all simulations below, we contrast SDN with a popular DNN framework, TensorFlow (Abadi et al., 2016). Additionally, we contrast DNN and SDN performances with a



(non-deep) instance-based learning model (IBL; a.k.a., case-based or exemplar-based) that employs a simple nearest-neighbor heuristic to select its outputs.<sup>4</sup> It is unclear whether the concept of IBL belongs more to the cognitive science or to the computer science community, as IBL models are widely used for both, modeling human cognition/behavior (e.g., Erickson & Kruschke, 1998; Gonzalez & Dutt, 2011; Gonzalez, Lerch, & Lebiere, 2003; Nosofsky, 1986, 1988, 1991) and finding efficient solutions for complex AI problems (Aha, Kibler, & Albert, 1991; Aha, 1991; Albert and Aha, 1991; Bareiss, 1991; Cover & Hart, 1967; Kibler, Aha, & Albert, 1989; Schlimmer, 1987; Wilson, 1997). Non-deep methods such as standalone IBL tend to suffer from noisy/irrelevant input features (although systems have been proposed to reduce this problem; e.g., Aha & Kibler, 1989). However, the purpose of including IBL here is not to promote deep methods over IBL, but rather to provide additional context—an additional baseline for evaluating SDN and DNN performance. In general, all model results provided here do not reflect optimal configurations of SDN, DNN, or IBL, but rather meaningful strengths and weaknesses of each system in addressing multi-attribute supervised classification problems.

For each of the simulations below, we employed datasets consisting of a number of input–output pairs (cases),  $N$ . To estimate the performance of each framework, we randomly shuffled the cases; split them up into training and test sets, where the size of the training set would be  $T$  and the size of the test set would be the remaining cases,  $N - T$ ; trained the given framework on the training cases; and recorded the proportion of correctly classified unseen test cases. This process was repeated twenty times, and the average correct classifications are taken as estimated performance metrics for each of the frameworks.

### 3.1. *Classification of numeric multi-attribute data*

For the first three simulations, we wanted to find out how SDN would perform on PITCHf/x, a dataset that is often included in TensorFlow tutorials. PITCHf/x is a dataset published by Major League Baseball Advanced Media, and it contains thousands of records of sensor data describing eight attributes of baseball pitches (three dimensions of ball velocity, three dimensions of ball acceleration, starting speed of the pitch, and whether the pitcher is right- or left-handed) along with seven corresponding pitch types (Fastball 2-seam, Fastball 4-seam, Fastball sinker, Fastball cutter, Slider, Changeup, Curveball). The goal for ML on this dataset is to learn how the sensor data map onto the pitch types. The downloaded PITCHf/x data included 7,700 cases.<sup>5</sup> We constructed the DNN model as per the suggested parameters in the tutorial (five layers, middle layer sizes being {250, 175, 150}). For SDN, the analog attributes in each case were converted to symbolic ones by binning and tiling (binning/tiling are long-standing methods in ML for converting analog to symbolic inputs; e.g., Sutton & Barto, 1998), where bin size was set to 10% of total attribute range and each next bin started half-way through the prior one (such that overlap between consecutive bins was 5% of total attribute range). Optimizing binning/tiling specifications for SDN and network sizes for DNN would undoubtedly lead to better results for current simulations, but again, the purpose here is not to find optimal results, but rather to provide evidence that symbolic cognitively inspired deep learning approaches show use and promise in the ML domain.

### 3.2. Efficiency

The TensorFlow tutorial suggests training on 7,000 cases and testing on the 700 remaining cases. However, DNNs require many more learning iterations than a few thousand cases would provide. To simulate bigger training data, DNN models are trained on the same training cases over multiple epochs. A major question that lacks a definitive answer is the following: what is the right number of epochs for training deep networks?

The answer depends largely on the training sample. A perfectly representative training sample can be looped over infinitely. Taking a lack of perfection as a given, training on the same sample too many times will lead to overfitting, where performance on training samples will not transfer to test samples. More formally, if  $f(x)$  is the performance of a DNN model on unseen data after being trained on some training data for  $x$  number of epochs, and  $0 < i < j < k$ , there exists some peak performance at  $j$  epochs, where  $f(i) < f(j) > f(k)$  regardless of values for  $i$  and  $k$ .

The problem is that  $j$  is an unknown quantity, and performance fluctuates up and down as the number of epochs is increased (i.e., just because performance decreased from 10 epochs to 20 epochs does not mean that 10 epochs is optimal—it could be a local maximum, and training for more epochs may bring performance backup). To examine DNN and SDN performance with different numbers of training epochs on PITCHf/x, we tested the frameworks by exponentially growing the number of epochs until performance stopped increasing. Such exponential growth until performance loss is not guaranteed to produce an optimal number of epochs, but it is a more robust stopping rule than the linear-growth alternative. Results from these simulations are shown in Figure 3, top.

Results indicate that SDN does not require more than 10 epochs to reach optimal performance, whereas DNN requires 1,000 (simulating millions of training examples where only thousands are provided). DNN optimal performance on this dataset is about 6% better than SDN. However, while SDN reaches 81% accuracy on this dataset within seconds, DNN GPU-optimized code requires over a minute to reach the same performance and another dozen minutes to improve another 6%.

This is highly encouraging given that DNNs popularity has resulted in better performance and faster runtimes over the course of the last few decades. We believe that as SDN popularity grows, the optimal performance gap between SDN and DNN can be closed, and SDN runtimes can be reduced even further as the code becomes GPU-optimized. Moreover, even at its current evolutionary stage, the cognitive approach seems to be a better choice than its AI counterpart for real-time applications where live/immediate learning and decisions are required. That is, when answers based on new data are needed within seconds, rather than minutes or hours, SDN can provide highly accurate classification estimates, within a few percentage points of eventual DNN performance.

### 3.3. Robustness to noise

The PITCHf/x dataset is fairly “clean” in the sense that there are no noisy and irrelevant input attributes. Unfortunately, algorithms like IBL are extremely sensitive to irrelevant and/or



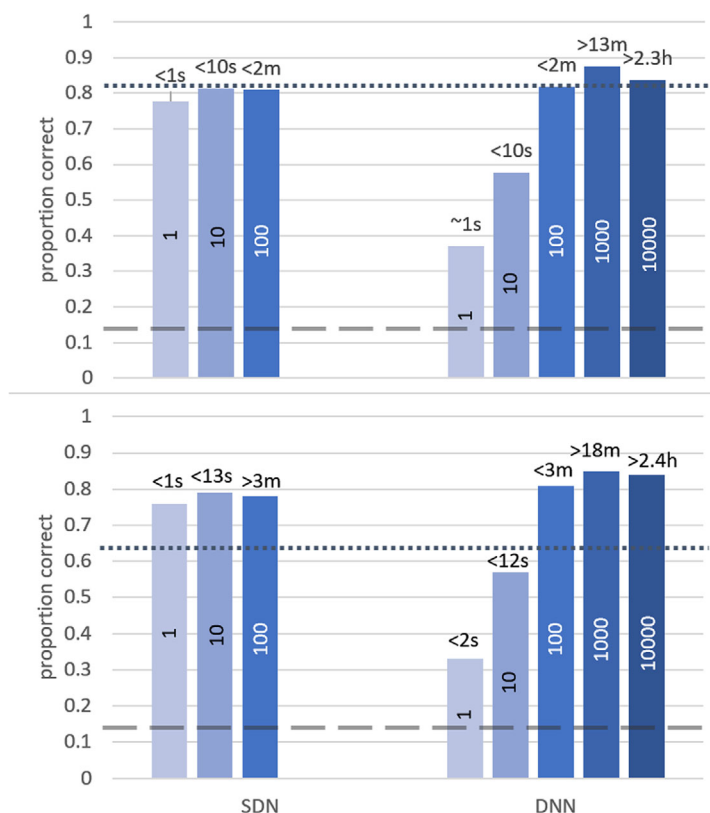


Fig. 3. SDN, DNN, and IBL performance on PITCHf/x dataset (700 test cases after training on 7,000 training cases), with the original attributes (top) and with four additional irrelevant/noise attributes (bottom). Number of training epochs for SDN and DNN is shown as vertical text inside respective columns. Time taken for training and testing a single model is shown as text above each respective column. IBL performance level is displayed as a dotted line. Random-level performance is 1 in 7, displayed as a dashed line. Standard error is negligible.

noisy dimensions (Aha et al., 1991). This is not the case for DNN. To examine whether SDN efficiency is affected by irrelevant attributes, as compared to DNN and IBL, we added four irrelevant attributes (four random numbers drawn from a uniform distribution) to each input vector in the PITCHf/x dataset and reran the simulation.

The average performance of each framework for PITCHf/x with four irrelevant attributes is displayed in Figure 3, bottom. As the results indicate, SDN and DNN performance drops about 1–2% when irrelevant attributes are added. This is not the case for IBL, where performance drops by nearly 20%. The conclusion we draw from this simulation is that despite being symbolic and lean, SDN retains the same robustness to noise that is found in sub-symbolic deep learning frameworks.

### 3.4. Catastrophic interference

An interesting phenomenon that is observed in DNN is catastrophic interference. Catastrophic interference is defined as significant loss of prior knowledge whenever new knowledge is acquired. A DNN agent will forget what it learned from a prior batch of examples when learning from new unrelated examples. This is the curse of fully distributed knowledge.

Of course, humans do not have this problem. Continuing with the PITCHf/x example, we can imagine learning about some pitch types prior to being exposed to others. For example, a baseball player in high school may be exposed to only five different pitch types. When they get to college, they are trained on the other two pitch types. We would expect this baseball player to perform poorly at recognizing the new pitch types when training starts, but after some exposures to the new pitches, they should perform well at categorizing all pitch types, rather than to forget how to categorize the original five pitch types. Learning-order effects may be present in human behavior, but they are not catastrophic.

To examine learning order effects in SDN and DNN, we shuffled the 7,700 PITCHf/x input–output pairs, split them into 7,000 training cases and 700 test cases, and then split the training data into two parts—early training data containing only five of the seven pitch types (Fastball 2-seam, Fastball 4-seam, Fastball sinker, Fastball cutter, and Slider), and late training data, containing the other two pitch types (Changeup, Curveball). There were about 5,000 training cases in the early training data and about 2,000 in the late training data. We trained SDN and DNN on early training data and tested both frameworks on the 700 test cases, expecting performance to be about 5:7 of the top performance from prior simulation. We then trained SDN and DNN on early data concatenated with the late data, and then tested the models on the 700 test cases. Essentially, this is a replication of the first simulation with 7,000 training and 700 test cases, with the only exception being the order in which training cases are presented. In the prior simulation, all pitch types were presented in random order, whereas in this case, the learners had to learn about five of the pitch types prior to learning about the other two pitch types.

Due to the well-known effects of catastrophic interference, we expected to see DNN performance plummet in this simulation, and SDN performance to be much more robust to these learning-order manipulations. Indeed, this is what we found (simulation results, averaged over 20 simulation runs, shown in Figure 4). IBL has no learning-order effects, so it provides a meaningful baseline here for absolute robustness to catastrophic interference. Note that SDN is not immune to learning-order effects, in that performance was lower in this simulation than in the first simulation. However, the reduction in performance is not catastrophic. More importantly, performance for SDN after training on late pitch data is *higher* than the performance after training on early pitch data, not lower, as is the case for DNN.

### 3.5. Common machine learning problem sets

To further assess SDN performance in the context of ML, we turned to common ML problem sets. We contrasted SDN with DNN and IBL on the four most popular (as of November, 2020) datasets from the UCI Machine Learning Repository (Dua & Graff, 2017), and the most

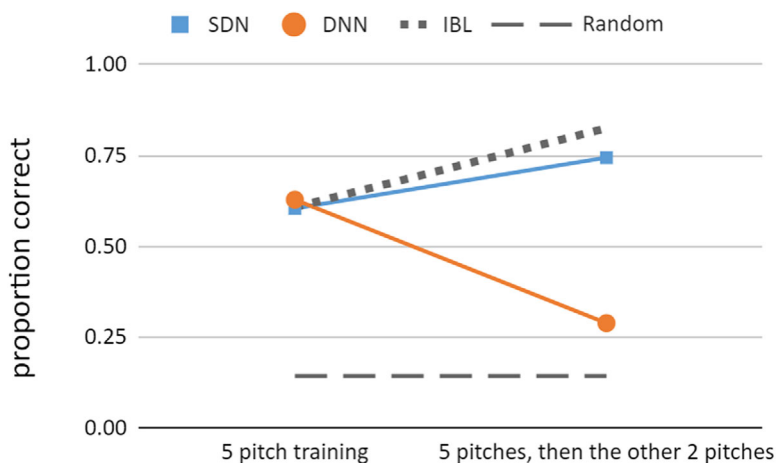


Fig. 4. SDN versus DNN performance on identifying seven different pitch types after training on just five of the pitches versus training on those same five pitches and then training on the remaining two pitches. Random-level performance is displayed as a dashed line. Standard error is negligible.

popular (as of November, 2020) dataset from [openml.org](https://openml.org). These five datasets were titled Iris,<sup>6</sup> Adult,<sup>7</sup> Wine,<sup>8</sup> Breast Cancer Wisconsin,<sup>9</sup> and credit-g.<sup>10</sup>

We used the same parameters for SDN and DNN frameworks as in the prior simulations, with the only difference being that DNN performance is reported for 100 epochs (rather than 1,000) for Adult and credit-g datasets, as this resulted in slightly better performance than that for 1,000 epochs. We used the same procedure as above to examine framework performance on each dataset—we shuffled each dataset, split it into training and test cases, trained on the training cases, and recorded accuracy at predicting the remaining unseen test cases (for the five datasets training:test sizes were 100:50, 30,000:2,561, 100:78, 600:99, and 900:100, respectively). This procedure was repeated 20 times, and the average performance for each framework on each dataset is reported in Figure 5.

Results indicate almost no difference between DNN and SDN performance across all five datasets. SDN reaches the same performance on the five datasets as DNN, with the exception of the Wine dataset, where DNN performance is 3 percentage points higher. Again, we find these results to be highly encouraging, given DNNs massive advantage in research and development allocation over the last couple of decades. Even at this evolutionary stage, SDN seems on par with DNN in many respects, while providing advantages in performance efficiency and robustness to catastrophic interference.

#### 4. Discussion

SDN work seems like a promising avenue of research. It is unclear at this point what SDN limitations are. It is certainly the case that the SDN framework examined here is not appropriate for image or sequence recognition, as it includes neither convolutional layers, nor

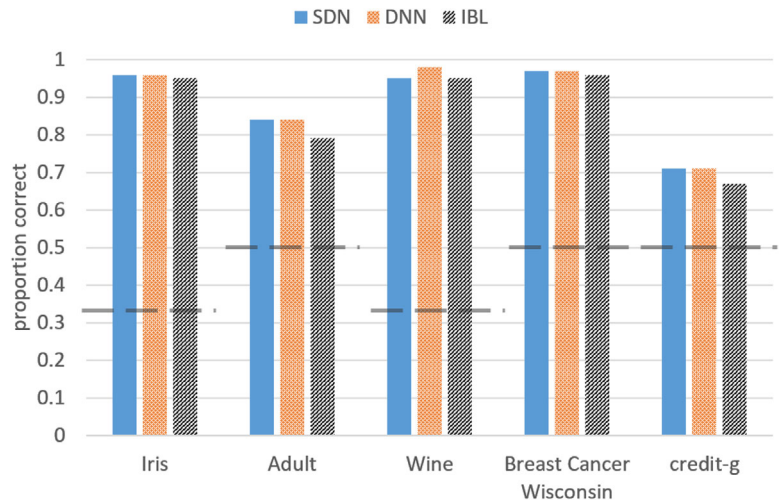


Fig. 5. SDN, DNN, and IBL performance on five of the most popular machine learning datasets. Random-level performance is displayed as a dashed line. Standard error is negligible.

recurrent connections. We find no reason to suspect that convolutional filters or recurrent connections are somehow incompatible with SDN. Moreover, we believe that symbolic pre-defined convolutional filters seem highly desirable, as these would require a small footprint and no expensive back-propagation.

Regardless of future SDN potential, it is fair to say that at the moment DNN methodology is far more appropriate than SDN for image and sequence recognition. There is also research suggesting that sub-symbolic fully-interconnected layers may be neuro-compatible. For example, neural nets are able to reproduce the type of “graceful” memory degradation that may be observed in old age or due to dementia (e.g., Rogers et al., 2004). It is unclear at this point to what degree SDNs are compatible with this gradual degradation phenomenon.

However, neuro-realism is neither a claim for DNNs, nor the eventual goal for SDN research. From the perspective of cognitive science, the goals of our research are more closely aligned with functional equivalence of human and animal cognition than with neuro-realism. From the perspective of artificial intelligence, the goal of SDN work is to improve on the fully interconnected sub-symbolic systems by borrowing mechanisms from cognitive modeling.

Finally, it may be possible that SDN- and DNN-type mechanisms could represent complementary systems in the brain. For example, McClelland, McNaughton, and O’Reilly (1995), acknowledging the problem of catastrophic interference in fully interconnected neural networks, suggested that a more sparsely connected neural network may serve as a complement to the denser neural net in the same way that the hippocampus serves as a complement to the cortex in the human brain. It may be possible that SDN could play a complementary role to DNN, making online learning more efficient, and aiding to avoid catastrophic interference.

To be clear, we do not believe that a purely sub-symbolic black-box fully-interconnected back-propagation approach to neural networks is the final theory and framework for simulating either the hippocampus or the cortex. In our opinion, as AI researchers begin to borrow more concepts from cognitive science, deep networks will become more lightweight, more explainable, and less prone to catastrophic interference. Perhaps, most importantly, they will become less reliant on large data, and will begin to approach human/animal learning efficiency.

## 5. Summary

In sum, we examined cognitively inspired symbolic deep nets in the ML domain across a number of simulations. We found that SDNs perform about as well as DNNs in most cases, but are much more lean and efficient. Like DNNs, SDNs are robust to noisy/irrelevant input attributes. Unlike DNNs, SDNs are also robust to catastrophic interference.

There were instances where, with enough training epochs, DNNs outperformed SDNs. Given enough data, enough computational resources, and enough time, DNNs will converge on optimal results. Deep neural nets are indeed magic—slow inefficient magic. What is encouraging, however, is that (1) even in cases where SDN performance is lower than DNN, SDNs converge on near-DNN-level accuracy very fast, and (2) as SDNs begin to receive more attention, research, and development, the gap between SDN and DNN accuracy may be reduced and erased.

It is important to note here that the reason why DNNs suffer from inefficiency and catastrophic interference is not necessarily because these networks are sub-symbolic, but rather because the knowledge in these networks is fully distributed. We hope that the current state of the art in DNNs is just the beginning, and that more attention will be given to *why* certain nodes should be connected to each other, rather than just relying on the law of large numbers to produce correct classification. In other words, we believe that the magic behind SDNs' high accuracy, efficiency, and robustness is not its symbolic nature, but rather its roots in cognitive science. To conclude, we want to suggest that more research and development in the AI community should focus on the *why* of cognitive functionality, rather than pure mathematical optimization and reliance on large data and high-end computational resources.

## Notes

- 1 Many computational memory networks comprise symbolic nodes connected by weighted (sub-symbolic) links, and are sometimes referred to as hybrid symbolic-sub-symbolic approaches. For the purpose of this paper, we will refer to all memory networks where nodes are explainable symbols as symbolic, including those that contain weighted links.
- 2 Catastrophic interference refers to a phenomenon where a trained DNN will begin to unlearn what it learned before if presented with new training data.

- 3 Given  $n$  features (e.g., *large*, *square*, *white*), we can create a chunk for every combination of feature presence and absence ( $\{large\}$ ,  $\{square\}$ ,  $\{white\}$ ,  $\{large, square\}$ ,  $\{large, white\}$ ,  $\{square, white\}$ , and  $\{large, square, white\}$ ). If we represent feature presence as a 1 and feature absence as a 0, we can represent each chunk as a binary number, and the total number of possible chunks is the total number of possible binary numbers, minus the blank chunk, which is  $2^n - 1$ . When each feature dimension can have two potential values, the total number of possible chunks is  $3^n - 1$ . With  $k$  possible values on  $n$  feature dimensions, we can have at most  $(k + 1)^n - 1$  possible chunks to represent all potential feature combinations.
- 4 Given a set of previously observed input vector instances,  $I$ , and corresponding correct outputs,  $O$ , and presented with a new input vector  $i$ , nearest-neighbor IBL model will find a previously observed input–output pair,  $i_x - o_x$ , such that  $i_x$  has the shortest geometric distance to  $i$  of all input vectors in  $I$ , and respond with  $o_x$ .
- 5 PITCHf/x data downloaded from the following two urls: [https://storage.googleapis.com/mlb-pitch-data/pitch\\_type\\_training\\_data.csv](https://storage.googleapis.com/mlb-pitch-data/pitch_type_training_data.csv)      [https://storage.googleapis.com/mlb-pitch-data/pitch\\_type\\_test\\_data.csv](https://storage.googleapis.com/mlb-pitch-data/pitch_type_test_data.csv)
- 6 <https://archive.ics.uci.edu/ml/datasets/Iris>
- 7 <https://archive.ics.uci.edu/ml/datasets/Adult>
- 8 <https://archive.ics.uci.edu/ml/datasets/Wine>
- 9 <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29>
- 10 <https://www.openml.org/d/31>

## References

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2016). TensorFlow: Large-scale machine learning on heterogeneous distributed systems. arXiv:1603.04467. Retrieved from tensorflow.org
- Aha, D. W. (1991). Case-based learning algorithms. In *Proceedings of the 1991 DARPA Case-Based Reasoning Workshop* (Vol. 1, pp. 147–158).
- Aha, D. W., & Kibler, D. F. (1989). Noise-tolerant instance-based learning algorithms. In *IJCAI* (pp. 794–799).
- Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1), 37–66.
- Albert, M. K., & Aha, D. W. (1991). Analyses of instance-based learning algorithms. In *AAAI* (pp. 553–558).
- Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Anderson, J. R. (2007). *How can the human mind exist in the physical universe?*. Oxford, England: Oxford University Press.
- Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum Associates Publishers.
- Bareiss, R. (1991). *Proceedings of the case-based reasoning workshop*. San Mateo, CA: Morgan Kaufmann, Inc.
- Byford, S. (2016). Google's alphago ai beats lee se-dol again to win go series 4-1. *The Verge*, 15.
- Chase, W. G., & Simon, H. A. (1973). Perception in chess. *Cognitive Psychology*, 4(1), 55–81.
- Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.
- d'Avila Garcez, A., Dutra, A. R. R., & Alonso, E. (2018). Towards symbolic reinforcement learning with common sense. *CoRR*, abs/1804.0. Retrieved from <http://arxiv.org/abs/1804.08597>

- Dua, D., & Graff, C. (2017). {UCI} Machine Learning Repository. Retrieved from <http://archive.ics.uci.edu/ml>
- Dutra, A. R. A., Garcez, A., & D'Avila Garcez, A. S. (2017). A comparison between deep Q-networks and deep symbolic reinforcement learning. In *CEUR Workshop Proceedings*. Retrieved from [http://ceur-ws.org/Vol-2003/NeSy17\\_paper6.pdf](http://ceur-ws.org/Vol-2003/NeSy17_paper6.pdf)
- Erickson, M. A., & Kruschke, J. K. (1998). Rules and exemplars in category learning. *Journal of Experimental Psychology: General*, 127(2), 107–140.
- Feigenbaum, E., & Simon, H. (1984). EPAM-like models of recognition and learning. *Cognitive Science*, 8(4), 305–336.
- Gluck, M. A., & Bower, G. H. (1988). From conditioning to category learning - an adaptive network model. *Journal of Experimental Psychology-General*, 117(3), 227–247.
- Gobet, F. (1998). Expert memory: A comparison of four theories. *Cognition*, 66(2), 115–152.
- Gobet, F. & Lane, P. C. R. (2005). The CHREST architecture of cognition: Listening to empirical data. In *Visions of Mind: Architectures for Cognition and Affect* (pp. 204–224).
- Gonzalez, C., & Dutt, V. (2011). Instance-based learning: Integrating sampling and repeated decisions from experience. *Psychological Review*, 118(4), 523–51. <https://doi.org/10.1037/a0024558>
- Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science*, 27(4), 591–635.
- Kibler, D., Aha, D. W., & Albert, M. K. (1989). Instance-based prediction of real-valued attributes. *Computational Intelligence*, 5(2), 51–57.
- Laird, J. E. (2012). *The Soar cognitive architecture*. Cambridge, MA: MIT Press.
- McClelland, J. L., McNaughton, B., & O'Reilly, R. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3), 419–457.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., & Others (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533.
- Nosofsky, R. M. (1986). Attention, similarity, and the identification-categorization relationship. *Journal of Experimental Psychology: General*, 115(1), 39–61.
- Nosofsky, R. M. (1988). Exemplar-based accounts of relations between classification, recognition, and typicality. *Journal of Experimental Psychology-Learning Memory and Cognition*, 14(4), 700–708.
- Nosofsky, R. M. (1991). Tests of an exemplar model for relating perceptual classification and recognition memory. *Journal of Experimental Psychology: Human Perception and Performance*, 17(1), 3–27.
- Nosofsky, R. M., Gluck, M. A., Palmeri, T. J., McKinley, S. C., & Glauthier, P. (1994). Comparing models of rule-based classification learning: A replication and extension of Shepard, Hovland, and Jenkins (1961). *Memory & Cognition*, 22(3), 352–369.
- Rogers, T. T., & McClelland, J. L. (2004). *Semantic cognition: A parallel distributed processing approach*. Cambridge, MA: MIT Press.
- Rusk, N. (2015). Deep learning. *Nature Methods*, 13(1), 1–35. <https://doi.org/10.1038/nmeth.3707>
- Schlimmer, J. C. (1987). Incremental adjustment of representations for learning. In *Proceedings of Fourth International Workshop on Machine Learning*, 79–90.
- Simon, H. A. (1967). Motivational and emotional controls of cognition. *Psychological Review*, 74, 29–39.
- Simon, H. A., & Feigenbaum, E. A. (1964). An information-processing theory of some effects of similarity, familiarization, and meaningfulness in verbal learning. *Journal of Verbal Learning and Verbal Behavior*, 3, 85–396. [https://doi.org/10.1016/S0022-5371\(64\)80007-4](https://doi.org/10.1016/S0022-5371(64)80007-4)
- Simon, H. A., & Gilmarin, K. (1973). A simulation of memory for chess positions. *Cognitive Psychology*, 5, 29–46. [https://doi.org/10.1016/0010-0285\(73\)90024-8](https://doi.org/10.1016/0010-0285(73)90024-8)
- Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: The MIT Press. <https://doi.org/10.1109/TNN.1998.712192>
- Veksler, V. D., & Buchler, N. (2019). Cognitive modeling with symbolic deep learning. In *17th Annual Meeting of the International Conference on Cognitive Modelling (ICCM 2019)*.



- Veksler, V. D., Buchler, N., LaFleur, C. G., Yu, M. S., Lebiere, C., & Gonzalez, C. (2020). Cognitive models in cybersecurity: Learning from expert analysts and predicting attacker behavior. *Frontiers in Psychology, 11*, 1049. <https://doi.org/10.3389/fpsyg.2020.01049>
- Veksler, V. D., Gluck, K. A., Myers, C. W., Harris, J., & Mielke, T. (2014). Alleviating the curse of dimensionality – A psychologically-inspired approach. *Biologically Inspired Cognitive Architectures, 10*, 51–60. <https://doi.org/10.1016/j.bica.2014.11.007>
- Wilson, D. R. (1997). *Advances in instance-based learning algorithms* (Unpublished doctoral dissertation). Cite-seer.
- Zhang, Q., & Sornette, D. (2017). *Learning like humans with deep symbolic networks*. arxiv.org. Retrieved from <http://arxiv.org/abs/1707.03377>