1. Introduction

Imagine a robot that can perform typical household tasks like cooking or laundry. Or one that can anticipate a dangerous maneuver before it takes place, preventing a traffic accident. Or one that can see a slippery surface and adjust accordingly. In fact, these already exist in research labs and they have a commonality: deep learning.

Deep learning is the science of training large artificial neural networks. Deep artificial neural networks (DANNs) can have hundreds of millions of parameters [1, 2], allowing them to model complex functions such as nonlinear dynamics. They form compact representations of state from raw, high-dimensional, multimodal sensor data commonly found in robotic systems to [3]. Unlike many machine learning methods, they do not require a human expert to hand-engineer feature vectors from sensor data at design time. DANNs can present particular challenges in physical robotic systems, where generating training data is generally expensive, and sub-optimal performance in training poses a danger in some applications. Yet, despite such challenges, roboticists are finding creative alternatives, such as leveraging training data via digital manipulation, automating training, and employing multiple DANNs to improve performance and reduce training time.

Applying deep learning to robotics is an active research area, with at least twenty-five papers published on the subject from 2014 through the time of this writing. This review presents a summary of this research with particular emphasis on the benefits and challenges vis-à-vis robotics. It is organized as follows: a brief primer on deep learning, a summary of recent research highlighting capabilities, an examination of limitations and strategies that mitigate these, and a conclusion containing future trends.

2. Deep learning

A brief history of deep learning

The basic principles of linear regression were used by Gauss and Legendre [5], and many of those same principles still cover what researchers in deep learning study. However, several important advances have slowly transformed regression into what we now call deep learning. First, the addition of an activation function enabled regression methods to fit to nonlinear functions. It also introduced some biological similarity with brain cells [6].

Next, nonlinear models were stacked in “layers” to create create powerful models, called *multi-layer perceptrons*. In the 1960s a few researchers independently figured out how to differentiate multi-layer perceptrons [7], and by the 1980s, it evolved into a popular method for training them, called backpropagation [8, 9]. It was soon proven that multi-layer perceptrons were universal function approximators [10], meaning they could fit to any data, no matter how complex, with arbitrary precision, using a finite number of regression units. In many ways, backpropagation marked the beginning of the deep learning revolution, however, researchers still mostly limited their neural networks to a few layers because of *the problem of vanishing gradients* [11, 12]. Deeper neural networks took exponentially longer to train.

In the 2000s, researchers began using graphical processing units (GPUs) to parallelize implementations of artificial neural networks [13]. The largest bottleneck in training neural networks is a matrix-vector multiplication step, which can be parallelized using GPUs. In 2006, Hinton presented a training method that he demonstrated to be effective with a many-layered neural network [14]. The near-simultaneous emergence of these technologies triggered the flurry of research interest that is now propelling deep learning forward at an unprecedented rate [15].

In the early years of artificial intelligence, the game of chess was considered representative of human intelligence over machines [18]. After machines beat world-class chess players [19], a new emblematic task was needed to represented the superior capabilities of human intelligence. Visual recognition was largely accepted to be something easy for humans but difficult for machines [20]. But now, with the emergence of deep learning, humans will not be able to claim that as an advantage for much longer. Deep learning has surged ahead of well-established image recognition techniques [21] and has begun to dominate the benchmarks in handwriting recognition [22], video recognition [23], small-image identification [24], detection in biomedical imaging [25-27], and many others. It has even achieved super-human accuracy in several image recognition contests [21, 28, 29].

Common DANN structures

DANNs are well-suited for use with robots because they can be used in structures that other machine learning models cannot support. Figure 1 diagrams four common structures for using DANNs with robots. Ultimately, the underlying philosophy that prevails in the deep learning community is that every part of a complex system can be made to "learn". Thus, the real power of deep learning does not come from using just one of these structures as a component in a robotics system, but in connecting parts of all of these structures together to form a full system that learns throughout. This is where the "deep" in deep learning begins to make its impact--when each part of a system is capable of learning, the system as a whole can adapt in sophisticated ways. To some extent, this idea requires the humans to be willing to relinquish a degree of control, but the benefit for doing so is that the system can then begin to learn on its own.

Structure A (in Figure 1) shows a DANN for regressing arbitrary functions. In this configuration, DANNs are typically trained using a "supervised learning" method, like stochastic gradient descent [17,59,61]. $\mathbf{x}$ and $\mathbf{y}$ can represent any known vectors of continuous values, and the DANN is used to estimate a function that maps from $\mathbf{x}$ to $\mathbf{y}$. This structure is highly useful in robotics, but is also generally well-understood, so we move on to some of the more interesting structures.

Structure B is called an autoencoder [14, 30]. It facilitates "unsupervised learning". It requires two DANNs, called an "encoder" and a "decoder". In this configuration, only $\mathbf{x}$ needs to be supplied by the user. $\mathbf{s}$ is a "latent" or internal encoding that the DANN generates. For example, $\mathbf{x}$ might represent images observed by a robot's camera, containing thousands or even millions of values. The encoder might use convolutional layers, which are known to be effective for digesting images [16,17,29,54]. $\mathbf{s}$ might be a small vector, perhaps only tens of values. By learning to reduce $\mathbf{x}$ to $\mathbf{s}$, the autoencoder essentially creates its own internal encoding of "state". It will not necessarily use an encoding that has meaning for humans, but it will be sufficient for the DANN to approximately reconstruct $\mathbf{x}$. How are autoencoders useful in robotics? Sometimes, the robot designer may not know exactly what values are needed by the robot. Autoencoders enable the system to figure that out autonomously. This becomes especially useful when a hybrid of supervised and unsupervised learning is used. For example, the user can impose certain values in $\mathbf{s}$ (perhaps, positional coordinates or joint angles) and the DANNs will learn to work with those values, using the other free elements in $\mathbf{s}$ for its own encoding purposes. Autoencoders may also be used to initialize some parts of structure C [53].

Structure C is a type of "recurrent neural network", which is designed to model dynamic systems, including robots. It is often trained with an approach called "backpropagation through time" [55,57]. Many advances, such as "long short term memory units" have made recurrent neural networks much stronger [56,21]. In this configuration, $\mathbf{u}$ represents a control signal. $\mathbf{s}$ is an internal representation of state. $\mathbf{x}$ is a vector of observed values. The transition function approximates how the control signal affects state over time. Just like with autoencoders, the representation of state can be entirely latent, or partially imposed by the user. (If it were entirely imposed, the model would be prevented from learning.) If $\mathbf{x}$ includes an estimate of the utility of state $\mathbf{s}$, then this configuration is used in "model-based reinforcement learning" [60].

Structure D learns a control policy. It can facilitate "model-free" reinforcement learning. It uses a DANN to evaluate the utility or quality, $\mathbf{q}$, of potential control vectors. $\mathbf{s}$ is a representation of state. $\mathbf{u}$ is a control vector. (Gradient methods can find the values for $\mathbf{u}$ that maximize $\mathbf{q}$. In cases with discrete control vectors, $\mathbf{u}$ may be omitted from the input-end and $\mathbf{q}$ augmented to contain an evaluation of each control vector.) Configurations like this are used when an objective task is known for the robot to perform, but the user does not know exactly how to achieve it. By rewarding the robot for accomplishing the task, it can be trained to learn how to prioritize its own choices for actions. As one prominent example, reinforcement learning was used to teach a machine to play a wide range of Atari video games [58].

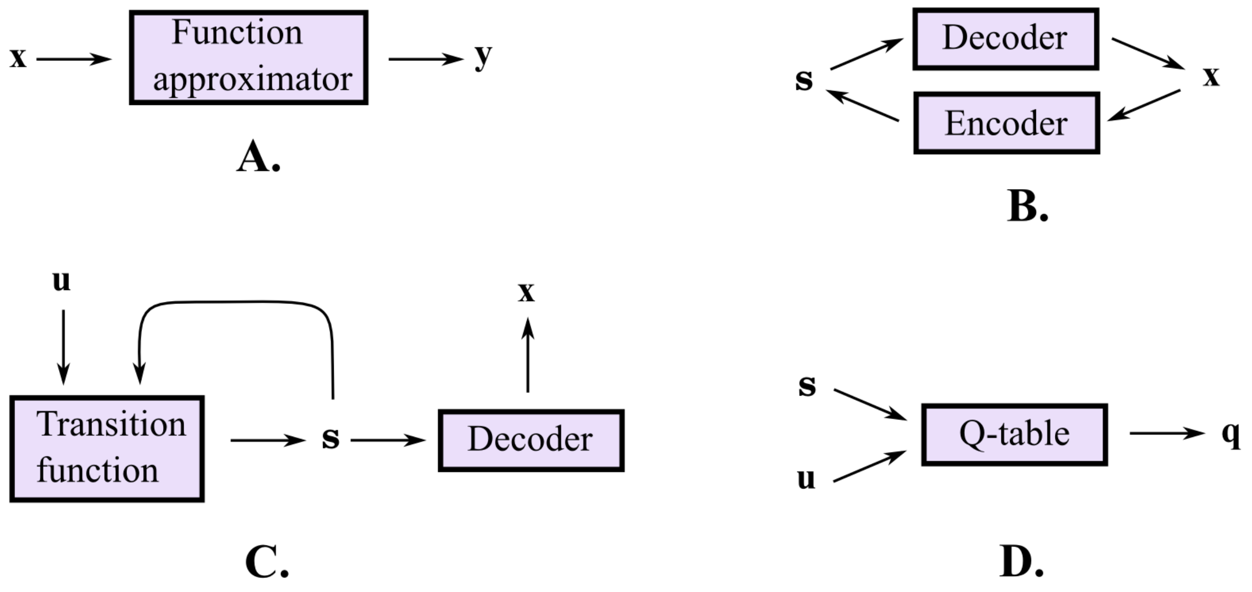

Figure 1: Diagram of some common structures for using neural networks with robots.

3. Recent research

Controlling complex dynamic systems

Many robotics problems involve controlling complex, time-varying dynamic systems, whether the dynamics of the robot itself or the dynamics of interactions between the robot and objects. Such problems often include hard-to-model nonlinear phenomena, variation and uncertainty in physical parameters, and/or system identification difficulties. Deep neural networks can learn these complex, nonlinear relationships, and offer the ability to control dynamic systems without explicit modeling.

Deducing system dynamics directly from full state information is one approach. Lenz, Knepper, and Saxena [30] researched robotic food cutting with a knife. They used deep learning to model the dynamics of the task (with a variation of structure C), which was in turn used to implement model predictive control (with a variation of structure D). The interaction between the knife and the food involves friction, deformation, hysteresis, etc., which are all difficult to model. The interface conditions also vary through the cut, as when the food-knife surface contact changes at the start of the cut, or the resistance changes as the knife passes through the peel to the center of a fruit. Their system outperformed fixed-trajectory stiffness control algorithms, increasing the mean cutting rate from 1.5 cm/s to 5.1 cm/s. Another example is Polydoros, Nalpantidis, and Kruger [31], who used deep learning to model the inverse dynamics of a manipulator. State variables obtained by operating under standard closed-loop control used to train the network. A “fading memory” feature allows the dynamic model to update quickly with changes to the robot dynamics caused by payload changes, mechanical wear, and other such issues. Analytically derived dynamic models have difficulty coping with such changes, and are very difficult to derive for highly compliant serial-elastic manipulators such as the new class of collaborative robots that have come on the market in the past four years. The authors report that their system out-performs the state-of-the-art in real-time learning evaluations and converges quickly, even in the presence of noisy sensor data. Punjani and Abbeel [32] used deep learning to model the dynamics of a radio-controlled helicopter to be used in model predictive control. The complex and highly coupled dynamics of helicopters are challenging to model analytically, and system identification is difficult. Instead a DANN was trained with a human expert flying the helicopter through various aerobatic maneuvers, and it outperformed three state-of-the-art methods for obtaining dynamic models of helicopters by about 60%.

Another advantage of DANNs is that they can be very computationally efficient at run-time, and they can be trained to control a dynamic system from partial state information [33]. Zhang, Kahn, and Levine [33] used deep learning to implement a model predictive control guided search for autonomous aerial vehicles. Reducing computational load in mobile robotics translates into power savings that increase range and/or improve performance. Without the need for full state information, fewer onboard sensors are required, saving cost and weight, and further reducing power consumption. check notes

DANNs also have the ability to control dynamic systems from video, without direct access to state information. Lillicrap, Hunt, and Pritzel [34] trained a DANN based on pixel data over 20 simulated dynamic systems, including the cart swing-up (inverted pendulum) problem, puck striking, and locomotion and developed a motion planning system that performed as well or better than algorithms that take advantage of the full state of the dynamic system, suggesting that many tasks can be learned from raw camera data. Finn, Levine, and Abbeel [35] used video of human experts performing a task to train a DANN to recognize nonlinear cost functions for motion planning. They demonstrated the ability to complete tasks that involved complex 2nd-order dynamics and hard-to-model interactions between a manipulator and various objects. They demonstrated their system on tasks tasks included 2D navigation, reaching, peg insertion, placing a dish, and pouring. mention inverse optimal control?

Object detection and manipulation

Detecting, grasping, and manipulating objects is another application in robotics where the ability to learn complex, nonlinear functions makes deep learning an attractive choice. An advantage of DANNs is their ability to operate on high-dimensional input data instead of requiring feature vectors to be hand-engineered at design time by experts who need expertise in both machine learning and the particular application [1]. This reduces dependence on human experts and the extra training time can at least be partially offset by the reduced up-front engineering effort.

DANNs have recently been applied to challenging object perception problems. Mariolis, Peleka, and Kargakos [36] studied object and pose recognition for textiles/garments hanging from a single point, as if picked by a robotic gripper. The system was trained on classes of objects (pants, shirts, and towels) of various size, shape, and material properties, as well as those hanging from various points. On a test set of six objects different from those used for training, the authors achieved 100% recognition and were able to predict pose, as measured by grasp point on the garment, with a mean error of 5.3 cm. These results were more accurate and faster than a baseline method based on support vector machines. Yang, Li, and Fermüller [37] trained a DANN to recognize 48 common kitchen objects and classify human grasps on those objects from 88 cooking videos from YouTube. Notably, these videos were not created with the purpose of training robots in mind. They exhibited significant variation in background and scenery. Grasps were classified as power or precision grasps, with subclasses of each for a total of six possible grasps. The authors achieved 79% accuracy on object recognition and 91% accuracy in classifying grasps. Chen, Qu, Zhou, Weng, Wang, and Fu [38] employed a convolutional neural network to identify the existence and pose of doors, and passed this information to a navigation algorithm for a mobile robot.

They suggest that navigating by such visual information can be superior to map-building–based methods in dynamic environments. Gao, Hendricks, Kuchenbecker, and Darrell [39] integrated both vison-based and contact-based perception to classify objects with haptic adjectives (smooth, compliant, etc.). If a robot can predict such information before making contact with the objects and quickly update its beliefs upon contact, it can quickly take appropriate actions such a adjusting its grip on a fragile object, avoiding slippery surfaces. They discuss the relative importance of visual and haptic data for the various objects studied. As with the other research studies in this paper, their method did not require manual design of feature vectors from domain-specific knowledge.

Other research has integrated object perception with grasp planning. Lenz, Lee, and Saxena [4] developed DANNs for grasp detection of objects from RGB-D (color + depth) images. The system does not require a 3D model of the object to be grasped and makes no assumptions about its geometry. In trials on Baxter and PR2 robots, successful grasps were selected 84% and 89% of the time, respectively. The authors compare this to a state-of-the art reference algorithm, which achieves only a 31% success rate. Yu, Weng, Liang, and Xie [40] employed deep learning to achieve object pose information from images. Objects were detected via dictionary-based methods, and once located, their pose was estimated and passed to a robotic manipulator for grasping. Pose configuration space was one dimensional (object on a plane) for the physical experiments and was discretized to facilitate classification with the DANN. The authors achieved a 95% success rate in correctly categorizing the pose for objects on which the DANN was trained.

Visuomotor control is an even closer integration between object detection and grasping, mapping image data directly to control signals applied to the manipulator's actuators. Levine, Finn, Darrell, and Abbeel [41] showed that such a system can be superior to designing and training separate systems for perception and control. Test applications included shape sorting, screwing a cap onto a bottle, fitting a hammer claw to a nail, and placing a coat hanger on a rack. The result was a system that could perform the tasks reliably, even when moderate visual distractors were present. Finn, Tan, Duan, Darrell, Levine, and Abbeel [42] also used a DANN, specifically a deep spatial auto-encoder (structure B) trained on visual features, to achieve visuomotor control. The autoencoder learned how robot actions affected the configuration of objects in the work envelope, and they were able to accomplish closed-loop control using these models. Tasks included pushing a block across a table, spooning material into a bowl, scooping a bag of rice off of a table with a spatula, and hanging a loop of rope on a hook. While this research may seem similar to visual servoing at first glance, it differs in that the control policies came from raw sensor data, and did not include any of the hand-designed feature vectors or transfer functions required for closed-loop control.

Scene understanding

Another important set of applications in the perception space is extracting meaning from scenes, either video or still images. Neverova, Wolf, Taylor, and Nebout [43] report on their first-place winning entry in the 2014 ChaLearn Looking At People Challenge (http://gesture.chalearn.org). This competition challenged entrants to automatically recognize 20 different Italian conversational gestures from 13,858 separate RGB-D videos of different people performing those gestures (only a limited subset of the data set is labeled for training). Hwang, Jung, Madapana, Kim, Choi, and Tani, [44] combined gesture recognition and object grasping functions into a DANN, demonstrating coordination between gesture recognition, attention switching, object perception, and grasping. Two objects were placed in front of the robot. The robot focused on a human collaborator, who gestured to indicate which object the robot was to pick up. The robot then switched its focus to the indicated object, recognized the object, and found an acceptable grasp. Their system was tested in simulation on an iCub humanoid robot, achieving a successful grasp 85% of the time. Ouyang and Wang [45] used a DANN to simultaneously address four independent aspects of pedestrian detection that are typically treated separately: feature extraction, articulation and motion handling, occlusion handling, and classification. They argue that their unified system avoids suboptimal interactions between these usually separate systems. The integrated network was trained on over 60,000 samples in the CalTech and ETH datasets. Compared to 18 other approaches in the published literature, the authors' system outperforms by 9% on the CalTech dataset and also outperforms on the ETH data set (no % improvement reported). Wu, Yildirim, Lim, Freeman, and Tenenbaum [46] attempted to predict the physical outcome of dynamic scenes by vision alone. This is based on the premise that humans can often look at a dynamic scene, predict parameters such as mass and friction from visual information alone, and then predict the outcome with some degree of accuracy. They use both simulations from a physics engine and physical trials involving a mass on an inclined plane.

Sensor fusion and dimensionality reduction

Another advantage of DANNs is that they are particularly good at handling the multimodal data generated in many robotic sensing applications. Examples include the integration of vision and haptic sensor data in Gao et al. [39] and the incorporation of depth data with pixel intensities as in the RGB-D image data from Lenz et al. [4] and Neverova et al. [43]. Additionally, Schmitz, Bansho, Noda, Iwata, Ogata, and Sugano [47] studied tactile object recognition with a *TWENDY-ONE* multi-finger hand, which provides data from distributed skin sensors, force and torque at the fingertips, actuator currents, and joint angles resulting in a multimodal set of 312 values. The system was trained on a set of twenty objects – some deliberately similar and some vastly different – handed to the robot in various poses that were unknown to the robot and grasped via a power grasp. The investigators achieved a success rate of 88% in recognizing objects handled by the robot, as compared to the 68% success rate using other methods in the literature. Jain, Koppula, Soh, Raghavan, Singh, and Saxena [48] trained a DANN to predict traffic maneuvers in a human-driven automobile. This was motivated by the fact that current collision avoidance systems often do not intervene in time to avoid an accident. Multimodal data inputs include video of the driver, video of the road in front of the car, dynamic state of the vehicle, GPS coordinates, and street maps of the area around the car. As with Neverova et al. [43], temporal dependencies complicate the problem. For instance, the time between a driver's head movement and the occurrence of a maneuver varies with vehicle speed. The system makes predictions every 0.8 seconds based on the last 5 seconds of data. The authors report a 90.5% accuracy rate in predicting maneuvers and can typically make the prediction about 3.5 seconds before the maneuver occurs.

DANNs have also shown the capability to form compact, low-dimensional representations from high-dimensional data [3]. Numerous examples that use still and/or video image data as direct inputs, without extracting hand-engineered feature vectors, have already been shown. Interestingly, Günther, Pilarski, Helfrich, Shen, and Diepold [49, 50] designed a DANN specifically to create meaningful feature vectors automatically. The research showed that deep learning was able to extract meaningful, low-dimensional features from high-dimensional camera images of welds from a laser welding process. These features were subsequently used with other machine learning and control strategies to close the loop on the welding process.

4. Challenges

For all of their benefits, DANNs do pose certain drawbacks. Perhaps the most significant in the field of robotics is the volume of training data required. While this poses a challenge in general, it is particularly problematic in robotics because generating training data on physical systems is often expensive. For instance, each simple task in Levine et al. [41] required 20-25 physical demonstrations. Jain et al. [48] trained their traffic maneuver prediction system on 1180 miles of high- and low-speed driving with 10 different drivers. Punjani and Abbeel [32] required the services of a human expert to repetitively demonstrate helicopter aerobatic maneuvers. Neverova et al. [43] had access to a set of over 13,000 videos of conversations, and Ouyang and Wang [45] had access to 60,000 samples for pedestrian detection. Compiling a similar data set for a new application may be time-consuming and expensive.

Despite this, the literature does contain clever approaches to mitigating this drawback. One approach is to generate virtual training data in simulation. To reduce the time and expense associated with generating physical training data, Mariolis et al. [36] pre-trained their networks on a large synthetic data set that was created in simulation using 3D graphics software. Wu et al. [46] trained, in part, on data simulated in a physics engine. Another tactic is leveraging training data through digital manipulation. Neverova et al. [43] faced the challenge that speed of conversational gestures varies significantly among different people. They varied the playback speed of their training videos to simulate this temporal variance, significantly expanding their training set without the need to acquire additional samples. Still other researchers utilizing reinforcement learning. Polydoros et al. [31] and Zhang et al. [33] used alternative control systems to automate the training of their networks.

Another challenge associated with the sheer size of DANNs is training time. Levine et al. [41], for instance, had 92,000 parameters in a 7-layer network, and reported training times of 3-4 hours for each task. This time investment is only practical for tasks that will be repeated frequently enough and often enough to provide an adequate payback. One solution reported in this group of studies is to distribute a given task among multiple, smaller DANNs. Mariolis et al. [36] trained separate DANNs for two subtasks in their garment pose recognition work. One DANN for object classification and passed passed its result to a different DANN trained to recognize pose. This multi-step approach sped both training and classification at run time. It should be noted, however, that this represents a tradeoff with efficiency of integrating separate tasks suggested by Levine et al. [41] and Ouyang and Wang [45]. Lenz et al. [30] distributed a single task amongst multiple networks when they employed a two-stage network design for grasp detection. The first DANN had fewer parameters, making it faster yet less accurate. It effectively eliminated highly unlikely grasps. The second stage had more parameters, making it more accurate, but it was relatively quick since it had to consider only the most likely grasps. They found the combination to be robust and computationally efficient.

The work of Zhang et al. [33] highlights two additional problems: First, unsupervised learning is not practical for robotic systems where a single failure is catastrophic, as in aerial vehicles. Second, the computational resources required for training a DANN are significant, and providing these in a system that is sensitive to weight, power consumption, and cost is often not practical. The authors' trained their aerial systems using a ground-based control system communicating with the vehicle wirelessly. This made training safe and automatic, and allowed them to use off-board computing resources for training.

Another challenge, articulated particularly accurately by Lenz, Lee, and Saxena [4], is that deep learning research has largely focused on classification tasks. While this is useful in many robotic perception tasks, many robotic tasks are concerned with finding optimal or near-optimal solutions within a continuous domain. For instance, one desires the grasp configuration that maximizes the likelihood of success [4] or the path that minimizes time or energy within given constraints.

5. Conclusion

Deep learning has shown promise in many significant problems in sensing, cognition, and action, and even the potential to combine these normally separate functions into a single system. DANNs can operate on raw sensor data and deduce key features in that data without human assistance, potentially reducing up-front engineering time, improving performance, or both. This also makes them adept at fusing high-dimensional, multimodal data. Improvement with experience has been demonstrated, allowing them to adapt to the dynamic, unstructured environments in which robots operate.

Training remains the most significant barrier to the wider use of deep learning in robotics. Generating training data on physical systems is often time consuming and expensive. One promising trend is cloud robotics, which offers the potential to crowdsource training data [51]. This data need not even be from other robots, as shown by Yang's use of general-purpose cooking videos for object and grasp recognition. Training data aside, the time required to train DANNs is still long enough to make them impractical for many robotic applications. Local parallel processing [13] and increases in raw processing speed have led to significant improvements. Distributed computing offers the potential to throw more computing resources at a given problem [51] but can be limited by bit-rate communication speeds [2]. Aside from applying brute-force computing power, there may be ways of making the training process more efficient. Deep learning researchers are currently working on directing the network's attention to the most relevant subspaces within the data for a given task. Further, they are looking into biologically inspired, sparsely connected DANNs, where each node has many short-range connections but few long-range connections, as is the case with neurons in the brain [21].

Much of the research presented deals with simple associative memory tasks such as object recognition. Some researchers believe that deep learning may well achieve human-level performance in such tasks in the near future [1, 51]. But researchers have not yet been able to understand the nature of 3D objects; "object invariance" is not captured [52]. It is not apparent at this point whether deep learning can conquer this challenge. Little research has been conducted in this direction, and cognitive training datasets do not even exist as of 2015 [51]. Pratt [51] summarizes the challenge as achieving "generalizable knowledge representation and cognition based on that representation." Whether this is achievable or how long it might take is unclear.

References

[1] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature,* vol. 521, pp. 436, 05/27, 2015.

[2] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science,* vol. 349, pp. 255-260, 07/17, 2015.

[3] W. Böhmer, J. T. Springenberg, J. Boedecker, M. Riedmiller and K. Obermayer, "Autonomous learning of state representations for control: An emerging field aims to autonomously learn state representations for reinforcement learning agents from their real-world sensor observations," *KI-Künstliche Intelligenz,* vol. 29, pp. 353-362, 2015.

[4] I. Lenz, H. Lee and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robotics Res.,* vol. 34, pp. 705-724, 04, 2015.

[5] S. M. Stigler, "Gauss and the invention of least squares," *The Annals of Statistics,* pp. 465-474, 1981.

[6] S. Haykin and N. Network, "A comprehensive foundation," *Neural Networks,* vol. 2, 2004.

[7] A. E. Bryson, W. F. Denham and S. E. Dreyfus, "Optimal programming problems with inequality constraints," *Aiaa J.,* vol. 1, pp. 2544-2550, 1963.

[8] Williams, DE Rumelhart GE Hinton RJ and G. Hinton, "Learning representations by back-propagating errors," *Nature,* vol. 323, pp. 533-536, 1986.

[9] P. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," 1974.

[10] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems,* vol. 2, pp. 303-314, 1989.

[11] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen Netzen," *Master's Thesis, Institut Fur Informatik, Technische Universitat, Munchen,* 1991.

[12] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems,* vol. 6, pp. 107-116, APR 1998, 1998.

[13] K. Oh and K. Jung, "GPU implementation of neural networks," *Pattern Recognit,* vol. 37, pp. 1311-1314, 2004.

[14] G. E. Hinton, S. Osindero and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.,* vol. 18, pp. 1527-1554, JUL 2006, 2006.

[15] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang and Q. V. Le, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems,* 2012, pp. 1223-1231.

[16] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The Handbook of Brain Theory and Neural Networks,* vol. 3361, pp. 1995, 1995.

[17] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems,* 2012, pp. 1097-1105.

[18] S. Thrun, "Learning to play the game of chess," *Advances in Neural Information Processing Systems,* vol. 7, 1995.

[19] M. Campbell, A. J. Hoane and F. Hsu, "Deep blue," *Artif. Intell.,* vol. 134, pp. 57-83, 2002.

[20] N. Pinto, D. D. Cox and J. J. DiCarlo, "Why is real-world visual object recognition hard?" *PLoS Comput Biol,* vol. 4, pp. e27, 2008.

[21] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks,* vol. 61, pp. 85-117, 01, 2015.

[22] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions On,* vol. 31, pp. 855-868, 2009.

[23] M. Yang, S. Ji, W. Xu, J. Wang, F. Lv, K. Yu, Y. Gong, M. Dikmen, D. J. Lin and T. S. Huang, "Detecting human actions in surveillance videos," in *TREC Video Retrieval Evaluation Workshop,* 2009.

[24] M. Lin, Q. Chen and S. Yan, "Network in network," *arXiv Preprint arXiv:1312.4400,* 2013.

[25] D. Ciresan, A. Giusti, L. M. Gambardella and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in Neural Information Processing Systems,* 2012, pp. 2843-2851.

[26] D. C. Cireşan, A. Giusti, L. M. Gambardella and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*Anonymous Springer, 2013, pp. 411-418.

[27] L. Roux, D. Racoceanu, N. Lomenie, M. Kulikova, H. Irshad, J. Klossa, F. Capron, C. Genestie, G. Le Naour and M. N. Gurcan, "Mitosis detection in breast cancer histological images An ICPR 2012 contest," *J. Pathol. Inform.,* vol. 4, pp. 8-3539.112693. Print 2013, May 30, 2013.

[28] D. Cireşan, U. Meier, J. Masci and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *Neural Networks (IJCNN), the 2011 International Joint Conference On,* 2011, pp. 1918-1921.

[29] D. Ciresan, U. Meier and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference On,* 2012, pp. 3642-3649.

[30] I. Lenz, R. Knepper and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control," in *Robotics Science and Systems (RSS),* 2015, .

[31] A. S. Polydoros, L. Nalpantidis and V. Kruger, "Real-time deep learning of robotic manipulator inverse dynamics," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference On,* 2015, pp. 3442-3448.

[32] A. Punjani and P. P. Abbeel, "Deep learning helicopter dynamics models," in *Robotics and Automation (ICRA), 2015 IEEE International Conference On,* 2015, pp. 3223-3230.

[33] T. Zhang, G. Kahn, S. Levine and P. Abbeel, "Learning Deep Control Policies for Autonomous Aerial Vehicles with MPC-Guided Policy Search," *arXiv Preprint arXiv:1509.06791,* 2015.

[34] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv Preprint arXiv:1509.02971,* 2015.

[35] C. Finn, S. Levine and P. Abbeel, "Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization," *arXiv Preprint arXiv:1603.00448,* 2016.

[36] I. Mariolis, G. Peleka, A. Kargakos and S. Malassiotis, "Pose and category recognition of highly deformable objects using deep learning," in *Advanced Robotics (ICAR), 2015 International Conference On,* 2015, pp. 655-662.

[37] Y. Yang, Y. Li, C. Fermüller and Y. Aloimonos, "Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web." in *Aaai,* 2015, pp. 3686-3693.

[38] W. Chen, T. Qu, Y. Zhou, K. Weng, G. Wang and G. Fu, "Door recognition and deep learning algorithm for visual based robot navigation," in *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference On,* 2014, pp. 1793-1798.

[39] Y. Gao, L. A. Hendricks, K. J. Kuchenbecker and T. Darrell, "Deep Learning for Tactile Understanding From Visual and Haptic Data," *arXiv Preprint arXiv:1511.06065,* 2015.

[40] Jincheng Yu, Kaijian Weng, Guoyuan Liang and Guanghan Xie, "A vision-based robotic grasping system using deep learning for 3D object recognition and pose estimation," in *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference On,* 2013, pp. 1175-1180.

[41] S. Levine, C. Finn, T. Darrell and P. Abbeel, "End-to-end training of deep visuomotor policies," *arXiv Preprint arXiv:1504.00702,* 2015.

[42] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine and P. Abbeel, "Deep Spatial Autoencoders for Visuomotor Learning," *Reconstruction,* vol. 117, pp. 240, 2015.

[43] N. Neverova, C. Wolf, G. W. Taylor and F. Nebout, "Multi-scale deep learning for gesture detection and localization," in *Computer Vision-ECCV 2014 Workshops,* 2014, pp. 474-490.

[44] Jungsik Hwang, Minju Jung, N. Madapana, Jinhyung Kim, Minkyu Choi and J. Tani, "Achieving "synergy" in cognitive behavior of humanoids via deep learning of dynamic visuo-motor-attentional coordination," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference On,* 2015, pp. 817-824.

[45] Wanli Ouyang and Xiaogang Wang, "Joint deep learning for pedestrian detection," in *Computer Vision (ICCV), 2013 IEEE International Conference On,* 2013, pp. 2056-2063.

[46] J. Wu, I. Yildirim, J. J. Lim, B. Freeman and J. Tenenbaum, "Galileo: Perceiving physical object properties by integrating a physics engine with deep learning," in *Advances in Neural Information Processing Systems,* 2015, pp. 127-135.

[47] A. Schmitz, Y. Bansho, K. Noda, H. Iwata, T. Ogata and S. Sugano, "Tactile object recognition using deep learning and dropout," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference On,* 2014, pp. 1044-1050.

[48] A. Jain, H. S. Koppula, S. Soh, B. Raghavan, A. Singh and A. Saxena, "Brain4Cars: Car That Knows Before You Do via Sensory-Fusion Deep Learning Architecture," *arXiv Preprint arXiv:1601.00740,* 2016.

[49] J. Günther, P. M. Pilarski, G. Helfrich, H. Shen and K. Diepold, "Intelligent laser welding through representation, prediction, and control learning: An architecture with deep neural networks and reinforcement learning," *Mechatronics*, 2015.

[50] J. Günther, P. M. Pilarski, G. Helfrich, H. Shen and K. Diepold, "First Steps Towards an Intelligent Laser Welding Architecture Using Deep Neural Networks and Reinforcement Learning," *Procedia Technology,* vol. 15, pp. 474-483, 2014.

[51] G. A. Pratt, "Is a Cambrian Explosion Coming for Robotics?" *Journal of Economic Perspectives,* vol. 29, pp. 51-60, Summer2015, 2015.

[52] J. Bohannon, "Helping Robots See the Big Picture," *Science,* vol. 346, pp. 186-187, 10/10, 2014.

[53] Gashler, Mike, and Tony Martinez. "Temporal nonlinear dimensionality reduction." *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011.

[54] LeCun, Yann, and Yoshua Bengio. "Convolutional networks for images, speech, and time series." *The handbook of brain theory and neural networks* 3361.10 (1995): 1995.

[55] Werbos, Paul J. "Backpropagation through time: what it does and how to do it." *Proceedings of the IEEE* 78.10 (1990): 1550-1560.

[56] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[57] Sjöberg, Jonas, et al. "Nonlinear black-box modeling in system identification: a unified overview." *Automatica* 31.12 (1995): 1691-1724.

[58] Mnih, Volodymyr, et al. "Playing Atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

[59] Dean, Jeffrey, et al. "Large scale distributed deep networks." *Advances in neural information processing systems*. 2012.

[60] Kuvayev, D., and Richard S. Sutton. *Model-based reinforcement learning*. Tech. rept. university of massachusetts, Dept of computer science, 1997.

[61] Wilson, D. Randall, and Tony R. Martinez. "The general inefficiency of batch training for gradient descent learning." *Neural Networks* 16.10 (2003): 1429-1451.