JCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Vol.-6, Special Issue-7, Sept. 2018 E-ISSN: 2347-2693

A Review on Deep Learning in Robotics

Shimi P S[1*] , Shajan P X[2]

[1,2]Sree Narayana Gurukulam College of Engineering, India

Available online at: www.ijcseonline.org

Abstract:-During the last few decades, there has been a rush in research in the area of deep learning. In this paper we have made a review on the limitations of deep learning in physical robotic systems, using currently available examples. It is mainly focused on the recent advances made in robotics community and application of deep learning in robotics.

Keywords: Deep neural networks; artificial intelligence; human-robot interaction

I. INTRODUCTION

Deep learning is the discipline of science involving training large artificial neural networks. Deep neural networks (DNNs) 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 [3], and unlike many machine learning methods, they do not require a human expert to hand-engineer feature vectors from sensor data at design time.

DNNs can, however, 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. However, despite such challenges, robotics is finding it difficult to create alternatives, such as leveraging training data via digital manipulation, automating training, and employing multiple DNNs to improve performance and reduce training time.

In this paper an introduction to deep learning is followed by a discussion on the limitations of deep learning and strategies that mitigate these as well as future trends are discussed.

II. DEEP LEARNING

1 A brief history of deep learning

A fundamental principles of linear regression were used by Gauss and Legendre in 1981 [4], and many of those same principles still cover what researchers in deep learning study. However, several 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 [5]. Next, the nonlinear models were stacked in "layers" to create powerful models, called *multi-layer perceptrons*. In the year 1960 a few researchers independently figured out how to differentiate multi-layer perceptrons [6], and by the 1980s, it evolved into a popular method for training them, called back propagation [7, 8]. It was soon proven that multi-layer perceptrons were universal function approximators [9], 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* [10, 11]. Deeper neural networks took exponentially longer to train.

Neural networks were successfully applied for robotics control in early 1980s [12]. It was quickly recognized that nonlinear regression provided the functionality that was needed for operating dynamical systems in continuous spaces [13, 14], and closely related fuzzy systems seemed well suited for nominal logical control decisions [15]. Even as early as 1989, Pomerleau's [16] famously demonstrated that neural networks were effective for helping vehicles to stay in their lanes. However, neural networks were still generally too slow to digest entire images, or perform the complex tasks necessary for many robotics applications.

In the year 2000, researchers began using graphical processing units (GPUs) to parallelize implementations of artificial neural networks [17]. 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 [18]. The near simultaneous emergence of these technologies triggered the flurry of research interest that is now propelling deep learning forward at an unprecedented rate [19].

As hardware improved, and as neural networks began to become more practical, they were increasingly found to be effective with real robotics applications. In 2004 RNNPB showed that neural networks could self-organize high-level control schema that generalized effectively with several robotics test problems [20]. In 2008, neuroscientists made advances in recognizing how animals achieved locomotion, and were able to extend this knowledge all the way to neural networks for experimental control of robots [21]. In 2011, TNLDR demonstrated that deep neural nets could effectively model both state and dynamics from strictly unsupervised training with raw images of a simulated robot [22]. Another relevant work is Pomerleau in 2012 in his book Surveying applications, for neural networks in perception for robot guidance [23].

In foresight, we see that chess was considered in the early years of artificial intelligence to be representative of human intelligence over machines [24]. After machines beat world-class chess players [25], a new emblematic task was needed to represent the superior capabilities of human intelligence. Visual recognition was largely accepted to be something easy for humans but difficult for machines [26]. 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 [27] and has begun to dominate the benchmarks in handwriting recognition [28], video recognition [29], small-image identification [30], detection in biomedical imaging [31-33], and many others. It has even achieved super-human accuracy in several image recognition contests [27, 34, 35]. Perhaps agility or dexterity will be a forthcoming achievement where machines will begin to demonstrate human like proficiency. If so, it appears that deep neural networks may be the learning model that enables it.

2. Common DNN structures

The main idea of using machine learning in controlling robots requires humans to be willing to relinquish a degree of control. This can seem counterintuitive at first, but the benefit for doing so is that the system can then begin to learn on its own. This makes the system capable of adapting, and therefore has potential to ultimately make better use of the direction that comes from humans.

DNNs are well suited for use with robots because they are flexible, and can be used in structures that other machine learning models cannot support. Figure 1 diagrams four common structures for using DNNs with robots.

Structure A described in Figure 1 shows a DNN for regressing arbitrary functions. It is typically trained by presenting a large collection of example training pairs:{ <**x**1, **y**1>,<**x**2, **y**2>, … , <**x***n*,**y***n*> }. An optimization method is applied to minimize the prediction loss. For regression problems, loss is typically measured with sum-squared error, and for classification problems it is often measured with crossentropy particularly when a softmax layer is used for the output layer of the neural network [36]. Traditionally, the most popular optimization method for neural networks is stochastic gradient descent [37], but improved methods such as RMSProp [38] and Adam [39] have recently garnered widespread usage. Some other considerations for training them effectively, once the training is finished, novel vectors may be fed in as **x** to compute corresponding predictions for **y**.

Structure B is called an auto encoder [40]. It is one common model for facilitating “unsupervised learning.” It requires two DNNs, called an “encoder” and a “decoder.” In this configuration, only **x** needs to be supplied by the user. **s** is a “latent” or internal encoding that the DNN generates. For example, **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 [35, 41, 42]. **s** might be a small vector, perhaps only tens of values. By learning to reduce **x** to **s**, the auto encoder 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 DNN to approximately reconstruct **x**.

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” [43, 44]. Many advances, such as “long short-term memory units,” have made recurrent neural networks much stronger [27, 45]. In this configuration, **u** represents a control signal. **u** may also contain recent observations. s is an internal representation of future state, and **x** is a vector of anticipated future observations. The transition function approximates how the control signal will affect state over time. Just as with auto encoders, 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 **x** includes an estimate of the utility of state **s**, then this configuration is used in “model-based reinforcement learning” [46].

Structure D learns a control policy. It can facilitate “model-free” reinforcement learning. It uses a DNN to evaluate the utility or quality, **q**, of potential control vectors. **s** is a representation of state, and **u** is a 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.

For example, reinforcement learning was used to teach a machine to play a wide range of Atari video games [47].

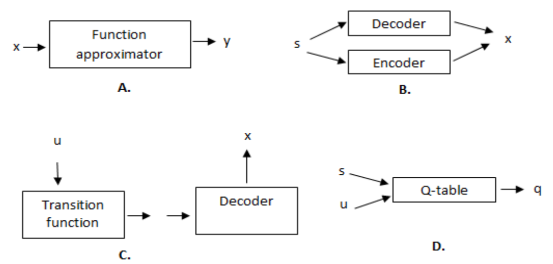

Figure 1. Diagram of some common structures for using neural networks with robots.

A: Function approximating models are trained to approximate the mappings represented in a training set of pair-wise examples. **B:** Auto encoders can reduce complex or high dimensional observations to a simple feature representation, often extracting intrinsic information from images. **C:** Recurrent models specialize in dynamics and temporal predictions. **D:** Policy models trained with reinforcement learning seek to plan the best decisions to make under possible future conditions.

III. DEEP LEARNING IN ROBOTICS

The robotics community has identified numerous goal for robotics in the next 5 to 20 years. These include, but certainly are not limited to, human-like walking and running, teaching by demonstration, mobile navigation in pedestrian environments, collaborative automation, automated bin/shelf picking, automated combat recovery, and automated aircraft inspection and maintenance, and robotic disaster mitigation and recovery [48- 52].

In this paper we have identified five general challenges for robotics that are critical for reaching these goals and for which DNN technology has high potential for impact:

Challenge 1: Learning complex, high-dimensional, and novel dynamics. Analytic derivation of complex dynamics requires human experts, is time consuming, and poses a trade-off between state dimensionality and tractability. Making such models robust to uncertainty is difficult, and full state information is often unknown. Systems that can quickly and autonomously adapt to novel dynamics are needed to solve problems such as grasping new objects, traveling over surfaces with unknown or uncertain properties, managing interactions between a new tool and/or environment, or adapting to degradation and/or failure of robot subsystems. Also needed are methods to accomplish this for systems that possess hundreds (or even thousands) of degrees of freedom, exhibit high levels of uncertainty, and for which only partial state information is available.

Challenge 2: Learning control policies in dynamic environments. As with dynamics, control systems that accommodate high degrees of freedom for applications such as multi-arm mobile manipulators, anthropomorphic hands, and swarm robotics are needed. Such systems will be called upon to function reliably and safely in environments with high uncertainty and limited state information.

Challenge 3: Advanced object recognition. DNNs have already proven to be highly adept at recognizing and classifying objects [27,34,35]. Advanced application examples include recognizing deformable objects and estimating their state and pose for grasping, semantic task and path specification (e.g., go around the table, to the car, and open the trunk), and recognizing the properties of objects and surfaces such as sharp objects that could pose a danger to human collaborators or wet/slippery floors.

Challenge 4: Interpreting and anticipating human actions. This challenge is critical if robots are to work with or amongst people in applications such as collaborative robotics for manufacturing, eldercare, autonomous vehicles operating on public thoroughfares, or navigating pedestrian environments. It will enable teaching by demonstration, which will in turn facilitate task specification by individuals without expertise in robotics or programming. This challenge may also be extended to perceiving human needs and anticipating when robotic intervention is appropriate.

Challenge 5: High-level task planning. Robots will need to reliably execute high-level commands that fuse the previous six challenges to achieve a new level of utility, especially if they are to benefit the general public. For example, the command “get the milk” must autonomously generate the lower-level tasks of navigating to/from the refrigerator, opening/closing the door, identifying the proper container (milk containers may take many forms), and securely grasping the container.

Loosely speaking, these challenges form a sort of “basis set” for the goals mentioned above. For example, human-like walking and running will rely heavily on Challenges 1 (learning dynamics) and 2 (learning control policies), while teaching by demonstration will require advances in Challenges 3 (object recognition) and 4 (interpreting human actions).

Table.1 categorizes recent robotics research that utilizes DNN technology according to these challenges, as well as the DNN structures discussed in the previous section. From this several observations are made: First is that Structure A is clearly the most popular DNN architecture in the recent robotics literature. This is likely explained by its intuitive nature, essentially learning to approximate the same function presented to it in the form of training samples. It also requires the least amount of domain knowledge in DNNs to implement. Robotics challenges, however, are not limited to the sort of classification and/or regression problems to which this structure is best suited.

Additional focus on applying Structures B, C, and D to robotics problems may very well catalyze significant advancement in many of the identified challenges. One of the purposes of this paper is to emphasize the potential of the other structures to the robotics community.

Somewhat related is the fact that some cells in Table 1 are empty. In the authors' opinion, this is due to a lack of research focus rather than any inherent incompatibilities between challenges and structures. In particular, the ability of Structure B to learn compact representations of state would be particularly useful for estimating the pose, state, and properties of objects (Challenge 4) and the state of human collaborators (Challenge 5).

Table 1. An overview of how DNN structures are used in the recent literature to address the five challenges

	DNN Structure			
Challenge	A	B	C	D
1 Dynamics	[76,78,81,85,115,127]	[87,112,113,115]	[115,122]	[125,126]
2 (Control)	[85,115]	[112]	[122]	[125,128]
3 (Manipulation)	[79,82-85,123]	[112]	[123]	[128]
4 (Human Actions)	[77,79,123,127]		[123]	
5 (High-level planning)				[128]

Table 1 also indicates limited application of DNNs to high-level task planning (Challenge 7). One of the barriers to the application of DNNs is quantifying the quality of such decisions. Standard benchmarks for decision quality are needed. Once this is addressed, DNNs may very well be able to be the tool that allows robotics to make progress on this very significant challenge.

The balance of this section is categorized by DNN structure and is organized as follows: 1) A discussion of the structure's role in robotics.
2) Examples from the recent literature of how the structure is being applied in robotics.
3) Practical recommendations for applying the structure in robotics.

IV. Classifiers and discriminative models (Structure in robotics

1 The role of Structure A in robotics

Structure A involves using a deep learning model to approximate a function from sample input-output pairs. This may be the most general-purpose deep learning structure, since there are many different functions in robotics that researchers and practitioners may want to approximate from sample observations. Some examples include mapping from actions to corresponding changes in state, mapping from changes in state to the actions that would cause it, or mapping from forces to motions. Whereas in some cases physical equations for these functions may already be known, there are many other cases where the environment is just too complex for these equations to yield acceptable accuracy. In such situations, learning to approximate the function from sample observations may yield significantly better accuracy.

The functions that are approximated need not be continuous. Function approximating models also excel at classification tasks, such as determining what type of object lies before the robot, which grasping approach or general planning strategy is best suited for current conditions, or what is the state of a certain complex object with which the robot is interacting.

The next section reviews some of the many applications for classifiers, regression models, and discriminative models that have appeared in the recent literature with robotics.

2 Generative and Unsupervised models (Structure B) in robotics

2.1 The role of Structure B in robotics

One of the characteristic capabilities that make humans so proficient at operating in the real world is their ability to understand what they perceive. A similar capability is offered in auto encoders, a type of deep learning model that both encodes observations into an internal representation, then decodes it back to the original observation. These models digest high-dimensional data and produce compact, low-dimensional internal representations that succinctly describe the meaning in the original observations [3].

Thus, auto-encoders are used primarily in cases where high-dimensional observations are available, but the user wants a low-dimensional representation of state. Generative models are closely related. They utilize only the decoding portion of an auto encoder, and are useful for predicting observations. Inference methods may be used with generative models to estimate internal representations of state without requiring an encoder to be trained at all. In many ways, generative models may be considered to be the opposite of classifiers, or discriminative models, because they map from a succinct representation to a full high-dimensional set of values similar to those that might typically be observed.

V. Policy learning models (Structure D) in robotics

1 The role of Structure D in robotics

Learning a near optimal (or at least a reasonably acceptable) control policy is often the primary objective in combining machine learning with robotics. The canonical model for using deep neural networks for learning a control policy is deep Q-learning [47]. It uses a DNN to model a table of Q-values, which are trained to converge to a representation of the values for performing each possible action in any state. Although Structure D is quite similar to Structure A in terms of the model itself, they are trained in significantly different ways. Instead of minimizing prediction error against a training set of samples, deep Q-networks seek to maximize long-term reward. This is done through seeking a balance between exploration and exploitation that ultimately leads to an effective policy model.

Ultimately, reinforcement learning models are useful for learning to operate dynamic systems from partial state information, and controllers based on deep reinforcement learning can be very computationally efficient at runtime [64]. They automatically infer priorities based on rewards that are obtained during training. In theory, they provide a complete control policy learning system, but they do suffer from extremely slow training times. Consequently, many of the works in the next section combine them with other approaches in order to seek greater levels of control accuracy and training speed.

VI. Current Shortcomings of DNNs for Robotics

For all of its benefits, deep learning does pose some drawbacks. Perhaps most significant is the volume of training data required, which is particularly problematic in robotics because generating training data on physical systems can be expensive and time consuming. For instance, Levine *et al.* [57] used 14 robots to collect over 800,000 grasp attempts over a period of 2 months. Jain *et al.* [62] had trained their traffic maneuver prediction system on 1180 miles of high- and low-speed driving with 10 different drivers. Punjani and Abbeel [53] required repeated demonstrations of helicopter aerobatic maneuvers by a human expert. Neverova *et al.* [54] had access to over 13,000 videos of conversations, and Ouyang and Wang [58] had access to 60,000 samples for pedestrian detection. Pinto and Gupta [65] needed 700 hours of robot time to generate a data set of 50,000 grasps for the training of a convolutional neural network. Despite this, the literature does contain clever approaches to mitigating this disadvantage. One approach entails using simulation to generate virtual training data.

For example, Mariolis *et al.* [55] pre-trained their garment pose recognition networks on a large synthetic data set created in simulation using 3D graphics software. Kappler *et al.* [66] generated a database of over 300,000 grasps on over 700 objects in simulation, generating physics-based grasp quality metrics for each and using this to classify grasp stability automatically. They validated via human classification of grasps and concluded that the computer- and human-generated labeling had good correlation. Another strategy is leveraging training data through digital manipulation.

Neverova *et al.* [54] faced the challenge that speed of conversational gestures varies significantly among different people. They varied video playback speed to simulate this temporal variance, expanding their training set without the need to acquire additional samples. Still other researchers utilizing reinforcement learning, such as Polydoros *et al.* [61] and Zhang *et al.* [64], automated training using alternative control systems during the learning phase.

Training time is another challenge associated with the sheer size of DNNs. Typical models involve up to millions of parameters and can take days to train on parallel hardware, which is practical only for frequently repeated tasks that provide adequate payback on training time invested. One way to reduce training time is distributing a task among multiple, smaller DNNs. Mariolis *et al.* [55] trained two DNNs: One performed object classification, and its result was passed to a second network for pose recognition. This multi-step approach sped both training and classification at runtime. Lenz *et al.* [63] employed a two-stage network design for grasp detection. The first DNN had relatively few parameters. Sacrificing accuracy for speed, it eliminated highly unlikely grasps. The second stage had more parameters, making it more accurate, but was relatively quick since it did not need to consider unlikely grasps. They found the combination to be robust and computationally efficient. It should be noted, however, that this strategy represents a tradeoff with other researchers' suggestions that integrating multiple functions within a single network results in better performance [58].

The work of Zhang *et al.* [64] highlights two additional challenges. First, unsupervised learning is not practical for robotic systems where a single failure is catastrophic, as in aerial vehicles. Second, providing the necessary computational resources for deep learning 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 wirelessly with the vehicle. This made training safe and automatic, and allowed them to use off-board computing resources for training.

VII. Conclusion

Deep learning has shown promise in significant sensing, cognition, and action problems, and even the potential to combine these normally separate functions into a single system. DNNs can operate on raw sensor data and deduce key features in that data without human assistance, potentially greatly reducing up-front engineering time. They are also adept at fusing high-dimensional, multimodal data. Improvement with experience has been demonstrated, facilitating adaptation in the dynamic, unstructured environments in which robots operate.

Some remaining barriers to the adoption of deep learning in robotics include the necessity for large training data and long training times. Generating training data on physical systems can be relatively time consuming and expensive. One promising trend is crowdsourcing training data via cloud robotics [67]. It is not even necessary that this data be from other robots, as shown by Yang's use of general-purpose cooking videos for object and grasp recognition [56]. Regarding training time, local parallel processing [17] and increases in raw processing speed have led to significant improvements. Distributed computing offers the potential to direct more computing resources to a given problem [60] but can be limited by communication speeds [2].

There may also be algorithmic ways of making the training process more efficient yet to be discovered. For example, deep learning researchers are actively working on directing the network's attention to the most relevant subspaces within the data and applying biologically inspired, sparse DNNs with fewer synaptic connections to train [27].

Ultimately, the trends are moving toward greater levels of cognition, and some researchers even believe that deep learning may achieve human-level abilities in the near future [1, 67]. However, deep learning still has many obstacles to overcome before achieving such an ambitious objective. Currently, cognitive training datasets do not even exist [67]. Although DNNs excel at 2D image recognition, they are known to be highly susceptible to adversarial samples [68], and they still struggle to model 3D spatial layouts with object invariance [65]. Currently, DNNs appear to be powerful tools for practitioners in robotics, but only time will tell whether they can really deliver the capabilities that are needed for dexterous adaptation in general environments.

REFERENCES

[1] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436-444.
[2] Jordan MI, Mitchell TM. Machine learning: arends, perspectives, and prospects. Science. 2015;349(6245):255-260.
[3] Böhmer W, Springenberg JT, Boedecker J, *et al.* 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. 2015;29(4):353-362.
[4] Stigler SM. Gauss and the invention of least squares. Ann of Statistics. 1981;9(3):465-474.
[5] Haykin S. Neural networks: a comprehensive foundation. 2nd ed. Upper Saddle River, New Jersey: Prentice Hall; 2004.
[6] Bryson AE, Denham WF, Dreyfus SE. Optimal programming problems with inequality constraints. AIAA Journal. 1963;1(11):2544-2550.
[7] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back propagating errors. Nature. 1986;323:533-536.
[8] Werbos P. Beyond regression: new tools for prediction and analysis in the behavioral sciences. [Ph.D. dissertation]. Dept. Statistics, Harvard Univ.; 1974.
[9] Cybenko G. Approximation by superpositions of a sigmoidal function. Math of Control, Signals and Sys. 1989;2(4):303-314.
[10] Hochreiter S. Untersuchungen zu dynamischen neuronalen netzen. [Master's thesis]. Institut Fur Informatik, Technische Universitat; 1991.
[11] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. of Uncertainty, Fuzziness and Knowledge-Based Syst. 1998;6(2).
[12] Miyamoto H, Kawato M, Setoyama T, & Suzuki R. Feedback-error-learning neural network for trajectory control of a robotic manipulator. Neural Networks 1998;1(3):251-265.
[13] Lewis FW, Jagannathan S, & Yesildirak A. (1998). Neural network control of robot manipulators and non-linear systems. CRC Press.
[14] Miller WT, Werbos PJ, & Sutton RS. (1995). Neural networks for control. MIT Press.
[15] Lin CT., & Lee CSG. Neural-network-based fuzzy logic control and decision system. IEEE Transactions on Computers. 1991;40(12):1320-1336.
[16] Pomerleau DA. (1989). ALVINN, an autonomous land vehicle in a neural network (No. AIP-77). Carnegie Mellon University, Computer Science Department.
[17] Oh K, Jung K. GPU implementation of neural networks. Pattern Recognition. 2004;37(6):1311-1314.
[18] Hinton GE, Osindero S, Teh Y. A fast learning algorithm for deep belief nets. Neural Computation. 2006;18(7):1527-1554.
[19] Dean J, Corrado G, Monga R, *et al.* Large scale distributed deep networks.Advances in Neural Information Process. Syst. 25; 2012.
[20] Tani J, Ito M, & Sugita Y. Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using RNNPB. Neural Networks. 2004;17(8):1273-1289.
[21] Ijspeert AJ. Central pattern generators for locomotion control in animals and robots: a review. Neural Networks. 2008;21(4):642-653.
[22] Gashler M, Martinez T. Temporal nonlinear dimensionality reduction. Neural Networks (IJCNN), 2011 International Joint Conference on; 2011. p. 1959-1966.
[23] Pomerleau DA (2012). Neural network perception for mobile robot guidance (Vol.239). Springer Science & Business Media.
[24] Thrun S. Learning to play the game of chess. Advances in Neural Inform. Process. Syst.: Proc. of the 1994 Conf.
[25] Campbell M, Hoane AJ, Hsu F. Deep blue. Artificial Intelligence. 2002;134(1):57-83.
[26] Pinto N, Cox DD, DiCarlo JJ. Why is real-world visual object recognition hard? PLoS Computational Biology. 2008;4(1).
[27] Schmidhuber J. Deep learning in neural networks: an overview. Neural Networks. 2015;6(1)85-117.
[28] Graves A, Liwicki M, Fernández S, *et al.* A novel connectionist system for unconstrained handwriting recognition. Pattern Anal and Machine Intell., IEEE trans. on. 2009;31(5):855-868.

[29] Yang M, Ji S, Xu W, *et al.* Detecting human actions in surveillance videos. TREC Video Retrieval Evaluation Workshop; 2009.
[30] Lin M, Chen Q, Yan S. Network in network. 2013. Available: https://arxiv.org/abs/1312.4400
[31] Ciresan D, Giusti A, Gambardella LM, *et al.* Deep neural networks segment neuronal membranes in electron microscopy images. Advances in Neural Information Processing Sys 25; 2012.
[32] Roux L, Racoceanu D, Lomenie N, *et al.* Mitosis detection in breast cancer histological images an ICPR 2012 contest. J Pathol Inform. 2013;4(8).
[33] Cireşan DC, Giusti A, Gambardella LM, *et al.* Mitosis detection in breast cancer histology images with deep neural networks. In: K. Mori, I. Sakuma, Y. Sato, C. Barillot and N. Navab, editors. Medical Image Computing and Computer- Assisted Intervention–MICCAI 2013. Springer; 2013.
[34] Cireşan D, Meier U, Masci J, *et al.* A committee of neural networks for traffic sign classification. Neural Networks (IJCNN), 2011 Int. Joint Conf. on; 2011. p. 1918-1921.
[35] Ciresan D, Meier U, Schmidhuber J. Multi-column deep neural networks for image
classification. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on; 2012. p. 3642-3649.
[36] Dunne RA, & Campbell NA. On the pairing of the softmax activation and crossentropy
penalty functions and the derivation of the softmax activation function.
In Proc. 8th Aust. Conf. on the Neural Networks, Melbourne 1997;181 (Vol.185).
[37] Wilson DR, Martinez TR. The general inefficiency of batch training for gradient descent learning. Neural Networks. 2003;16(10):1429-1451.
[38] Tieleman T, Hinton G. (2012). Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 4(2).
[39] Kingma D, Ba J. (2014). Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[40] Vincent P, Larochelle H, Lajoie I, *et al.* Stacked denoising dutoencoders: learning useful representations in a deep network with a local denoising criterion. J Mach Learning Research. 2010;11:3371-3408.
[41] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25; 2012.
[42] LeCun Y, Bengio Y. Convolutional networks for images, speech, and time series. In M. Arbib, editor. The Handbook of Brain Theory and Neural Networks. 2nd edition. Cambridge, MA: MIT Press; 2003.
[43] Werbos PJ. Backpropagation through time: what it does and how to do it. Proc. IEEE. 1990;78(10):1550-1560.
[44] Sjöberg J, Zhang Q, Ljung L, *et al.* Nonlinear black-box modeling in system identification: a unified overview. Automatica. 1995;31(12):1691-1724.
[45] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997;9(8):1735-1780.
[46] Atkeson CG, Santamaria JC. A comparison of direct and model-based reinforcement learning. Robotics and Automation, IEEE Int. Conf. on; Albuquerque, NM. 1997. p. 3557-3564.
[47] Mnih V, Kavukcuoglu K, Silver D, *et al.* Human-level control through deep reinforcement learning. Nature. 2015;518:529-533.
[48] A roadmap for US robotics: from internet to robotics, 2016 edition.
[49] World Technology Evaluation Center, Inc. International Assessment of Research and Development in Robotics. Baltimore, MD, USA; 2006.
[50] FY2009-2034 Unmanned systems integrated roadmap. Washington, DC: Department of Defence (US); 2009.
[51] Material Handling Institute. Material handling and logistics U.S. roadmap 2.0. 2017.
[52] DARPA Robotics Challenge [Internet]. [cited 2017 May 20]. Available from: http://www.darpa.mil/program/darpa-robotics-challenge
[53] Punjani AP, Abbeel P. Deep learning helicopter dynamics models. Robotics and Automation (ICRA), 2015 IEEE International Conference on; 2015. p. 3223- 3230.
[54] Neverova N, Wolf C, Taylor GW, *et al.* Multi-scale deep learning for gesture detection and localization. Computer Vision-ECCV 2014 Workshops; 2014. p. 474-490.
[55] Mariolis I, Peleka G, Kargakos A, *et al.* Pose and category recognition of highly deformable objects using deep learning. Advanced Robotics (ICAR), 2015 International Conference on; Istanbul. 2015. p. 655-662.
[56] Yang Y, Li Y, Fermüller C, *et al.* Robot learning manipulation action plans by watching unconstrained videos from the world wide web. 29th AAAI Conference on Artificial Intelligence (AAAI-15); Austin, TX. 2015.
[57] Levine S, Pastor P, Krizhevsky A, *et al.* Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. 2016. Available: http://arxiv.org/abs/1603.02199
[58]Ouyang W, Wang X. Joint deep learning for pedestrian detection. Computer Vision, 2013 IEEE Int. Conf. on; Sidney, VIC. 2013. p. 2056-2063.
[59] Wu J, Yildirim I, Lim JJ, *et al.* Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. Advances in Neural Information Processing Systems 28; 2015.
[60] Schmitz A, Bansho Y, Noda K, *et al.* Tactile object recognition using deep learning and dropout. Humanoid Robots, 2014 14th IEEE-RAS Int. Conf. on; 2014. p. 1044-1050.
[61]Polydoros AS, Nalpantidis L, Kruger V. Real-time deep learning of robotic manipulator inverse dynamics. Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on; 2015. p. 3442-3448.
[62]Jain A, Koppula HS, Soh S, *et al.* Brain4Cars: car that knows before you do via sensory-fusion deep learning architecture. 2016. Available: http://arxiv.org/abs/1601.00740
[63]Lenz I, Knepper R, Saxena A. Deepmpc: learning deep latent features for model predictive control. Robotics: Science and Systems XI; Rome, Italy. 2015.
[64]Zhang T, Kahn G, Levine S, *et al.* Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search. 2015. Available: http://arxiv.org/abs/1509.06791
[65]Pinto L, Gupta A. Supersizing self-supervision: learning to grasp from 50k tries and 700 robot hours. 2015. Available: http://arxiv.org/abs/1509.06825
[66]Kappler D, Bohg J, Schaal S. Leveraging big data for grasp planning. 2015 IEEE International Conference on Robotics and Automation (ICRA); Seattle, WA. 2015. p. 4304-4311.
[67] Pratt GA. Is a cambrian explosion coming for robotics? Journal of Economic Perspectives. 2015;29(3):51-60.
[68] Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks. 2013. Available: http://arxiv.org/abs/1312.6199