iScience

CellPress
OPEN ACCESS

Review

Integrating Machine Learning with Human Knowledge

Changyu Deng,[1] Xunbi Ji,[1] Colton Rainey,[1] Jianyu Zhang,[1] and Wei Lu[1,2,*]

SUMMARY

Machine learning has been heavily researched and widely used in many disciplines. However, achieving high accuracy requires a large amount of data that is sometimes difficult, expensive, or impractical to obtain. Integrating human knowledge into machine learning can significantly reduce data requirement, increase reliability and robustness of machine learning, and build explainable machine learning systems. This allows leveraging the vast amount of human knowledge and capability of machine learning to achieve functions and performance not available before and will facilitate the interaction between human beings and machine learning systems, making machine learning decisions understandable to humans. This paper gives an overview of the knowledge and its representations that can be integrated into machine learning and the methodology. We cover the fundamentals, current status, and recent progress of the methods, with a focus on popular and new topics. The perspectives on future directions are also discussed.

INTRODUCTION

Machine learning has been heavily researched and widely used in many areas from object detection (Zou et al., 2019) and speech recognition (Graves et al., 2013) to protein structure prediction (Senior et al., 2020) and engineering design optimization (Deng et al., 2020; Gao and Lu, 2020; Wu et al., 2018). The success is grounded in its powerful capability to learn from a tremendous amount of data. However, it is still far from achieving intelligence comparable to humans. As of today, there have been few reports on artificial intelligence defeating humans in sensory tasks such as image recognition, object detection, or language translation. Some skills are not acquired by machines at all, such as creativity, imagination, and critical thinking. Even in the area of games where humans may be beaten, machine behaves more like a diligent learner than a smart one, considering the amount of data requirement and energy consumption. What is worse, pure data-driven models can lead to unintended behaviors such as gradient vanishing (Hu et al., 2018; Su, 2018) or classification on the wrong labels with high confidence (Goodfellow et al., 2014). Integrating human knowledge into machine learning can significantly reduce the data required, increase the reliability and robustness of machine learning, and build explainable machine learning systems.

Knowledge in machine learning can be viewed from two perspectives. One is "general knowledge" related to machine learning but independent of the task and data domain. This involves computer science, statistics, neural science, etc., which lays down the foundation of machine learning. An example is the knowledge in neural science that can be translated to improving neural network design. The other is "domain knowledge" which broadly refers to knowledge in any field such as physics, chemistry, engineering, and linguistics with domain-specific applications. Machine learning algorithms can integrate domain knowledge in the form of equations, logic rules, and prior distribution into its process to perform better than purely data-driven machine learning.

General knowledge marks the evolution of machine learning in history. In 1943, the first neuron network mathematical model was built based on the understanding of human brain cells (McCulloch and Pitts, 1943). In 1957, perceptron was invented to mimic the "perceptual processes of a biological brain" (Rosenblatt, 1957). Although it was a machine instead of an algorithm as we use today, the invention set the foundation of deep neuron networks (Fogg, 2017). In 1960, the gradients in control theory were derived to optimize the flight path (Kelley, 1960). This formed the foundation of backpropagation of artificial neural networks. In 1989, Q-learning was developed based on the Markov process to greatly improve the

[1]Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA

[2]Department of Materials Science & Engineering, University of Michigan, Ann Arbor, MI 48109, USA

*Correspondence: weilu@umich.edu

https://doi.org/10.1016/j.isci.2020.101656

practicality and feasibility of reinforcement learning (Watkins and Dayan, 1992). In 1997, the concept of long short-term memory was applied to a recurrent neural network (RNN) (Hochreiter and Schmidhuber, 1997). The development of these algorithms, together with an increasing amount of available data and computational power, brings the era of artificial intelligence today.

Domain knowledge plays a significant role in enhancing the learning performance. For instance, experts' rating can be used as an input of the data mining tool to reduce the misclassification cost on evaluating lending applications (Sinha and Zhao, 2008). An accurate estimation of the test data distribution by leveraging the domain knowledge can help design better training data sets. Human involvement is essential in several machine learning problems, such as the evaluation of machine-generated videos (Li et al., 2018). Even in areas where machine learning outperforms humans, such as the game of Go (Silver et al., 2017), learning from records of human experience is much faster than self-play at the initial stage.

Knowledge is more or less reflected in all data-based models from data collection to algorithm implementation. Here, we focus on typical areas where human knowledge is integrated to deliver superior performance. In Section Knowledge and Its Representations, we discuss the type of knowledge that has been incorporated in machine learning and its representations. Examples to embed such knowledge will be provided. In Section Methods to Integrate Human Knowledge, we introduce the methodology to incorporate knowledge into machine learning. For a broad readership, we start from the fundamentals and then cover the current status, remarks, and future directions with particular attention to new and popular topics. We do not include the opposite direction, i.e. improving knowledge-based models by data-driven approaches. Different from a review on related topics (Rueden et al., 2019), we highlight the methods to bridge machine learning and human knowledge, rather than focusing on the topic of knowledge itself.

KNOWLEDGE AND ITS REPRESENTATIONS

Knowledge is categorized into general knowledge and domain knowledge as we mentioned earlier. General knowledge regarding human brains, learning process, and how it can be incorporated is discussed in Section Human Brain and Learning. Domain knowledge is specifically discovered, possessed and summarized by experts in certain fields. In some subject areas, domain knowledge is abstract or empirical, which makes it challenging to be integrated into a machine learning framework. We discuss some recent progresses on this form of knowledge in Section Qualitative Domain Knowledge. Meanwhile, the knowledge base is becoming more systematic and quantitative in various fields, particularly in science and engineering. We discuss how quantitative domain knowledge can be utilized in Section Quantitative Domain Knowledge.

Human Brain and Learning

Machine learning uses computers to automatically process data to look for patterns. It mimics the learning process of biological intelligence, especially humans. Many breakthroughs in machine learning are inspired by the understanding of learning from fields such as neuroscience, biology, and physiology. In this section, we review some recent works that bring machine learning closer to human learning.

For decades, constructing a machine learning system required careful design of raw data transformation to extract their features for the learning subsystem, often a classifier, to detect or classify patterns in the input. Deep learning (LeCun et al., 2015) relaxes such requirements by stacking multiple layers of artificial neural modules, most of which are subject to learning. Different layers could extract different levels of features automatically during training. Deep learning achieved record-breaking results in many areas. Despite dramatic increase in the size of networks, the architecture of current deep neural networks (DNNs) with 10^7 learnable weights is still much simpler than the complex brain network with 10^{11} neurons (Herculano-Houzel, 2009) and 10^{15} synapses (Drachman, 2005).

Residual neural networks (ResNets) (He et al., 2016), built on convolutional layers with shortcuts between different layers, were proposed for image classification. The input of downstream layers also contains the information from upstream layers far away from them, in addition to the output of adjacent layers. Such skip connections have been observed in brain cells (Thomson, 2010). This technique helps ResNets to win first place on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2015 classification task (He et al., 2016) and is widely used in other DNNs (Wu et al., 2019).

Dropout, motivated by "theory of the role of sex in evolution", is a simple and powerful way to prevent overfitting of neural networks (Hinton et al., 2012; Srivastava et al., 2014). It is also analogous to the fact that neurons and connections in human brains keep growing and dying (breaking). At the training time, the connections of artificial neurons break randomly, and the remaining ones are scaled accordingly. This process can be regarded as training different models with shared weights. At the test time, the outputs are calculated by a single network without dropout, which can be treated as an average of different models. In recent years, dropout is proposed to be used at the test time to evaluate uncertainty (Gal and Ghahramani, 2016; Gal et al., 2017).

The focus of machine learning nowadays is on software or algorithms, yet some researchers start to redesign the hardware. Current computers, built on the von Neumann architecture, have a power consumption that is several orders of magnitude higher than biological brains. For example, International Business Machines Corporation (IBM)'s Blue Gene/P supercomputer simulated 1% of the human cerebral cortex and could consume up to 2.9 MW of power (Hennecke et al., 2012) while a human brain consumes only around 20 W (Modha, 2017). Neuromorphic systems were proposed and designed to improve energy efficiency. To name a few practical implementations, TrueNorth from IBM (Merolla et al., 2014; Modha, 2017) can solve problems from vision, audition, and multi-sensory fusion; Loihi from Intel (Davies et al., 2018) can solve the least absolute shrinkage and selection operator (LASSO) optimization orders of magnitude superior than the conventional central processing unit (CPU); NeuroGrid from Stanford University (Benjamin et al., 2014) offers affordable biological real-time simulations. In these brain-inspired systems, information is stored and processed at each neuron's synapses, which themselves are responsible for learning (Bartolozzi and Indiveri, 2007). There are various models to mimic neurons by circuits, such as the integrate-and-fire (I&F) model (Bartolozzi and Indiveri, 2007), the Hodgkin-Huxley model (Hodgkin and Huxley, 1952), and the Boolean logic gate design (Deshmukh et al., 2005). The majority of implementations use the I&F model, though their specific circuitry and behaviors vary (Nawrocki et al., 2016). As for materials, most systems utilize the standard silicon fabrication, which allows for integration with relatively mature technology and facilities. Recently, a few neuromorphic components or systems are made of new materials, such as organic or nanoparticle composites (Sun et al., 2018; Tuchman et al., 2020).

Qualitative Domain Knowledge

Along with the neuroscience-inspired fast development of deep learning architecture, knowledge from specific domains enlightens innovative machine learning algorithms. In its primitive format, knowledge is descriptive, intuitive, and often expressed by plain language. Such knowledge is usually based on observation, inference, and induction. Here, we group it as qualitative domain knowledge which requires intensive engagement and interpretation by humans with sophisticated skills. Although there is no universal way to bridge machine learning and qualitative knowledge, qualitative knowledge adds unique insights into the machine learning framework by using customized strategies. This integration offers two primary advantages. For one thing, qualitative knowledge is explained mainly by experts, which means that it highly relies on subjective interpretation. Machine learning consolidates the stringency of expert knowledge so that it can be directly validated by the large amount of raw data. Therefore, the qualitative knowledge can be built on a more statistically rigorous base. For another thing, machine learning models, such as the DNN, are subject to interpretability issues. Using qualitative domain knowledge in machine learning helps dig into the underlying theoretical mechanisms.

Qualitative knowledge can be further divided into three subgroups according to the degree of quantification. Here, we name them as *Knowledge in Plain Language*, *Loosely Formed Knowledge*, and *Concretely Formatted Knowledge*.

Knowledge in Plain Language

Tremendous qualitative knowledge is well established in different disciplines, especially for social science. Sociology has been developed for thousands of years, with modern theory focusing on globalization and micro-marco structures (Ritzer and Stepnisky, 2017). Political scientists proposed institutional theories to profile current governments (Peters, 2019). These theories are usually in the form of plain language. Traditionally, machine learning is far from these domains. However, many empirical theories actually provide good intuition to understand and design better machine learning models. For example, machine learning researchers found that some widely used DNN models have "shape bias" in word recognition (Ritter et al., 2017). This means that the shape of characters is more important than color or texture in visual recognition.

At the same time, research in development psychology shows that humans tend to learn new words based on similar shape, rather than color, texture, or size (Nakamura et al., 2012a). This coincidence provides a new theoretical foundation to understand how the DNN identifies objects. Conversely, some unfavorable biases, such as those toward race and gender, should be eliminated (DeBrusk, 2018).

Loosely Formed Knowledge

Qualitative knowledge can be pre-processed in ways that it is expressed in more numerical formats for use in machine learning. One example is that empirical human knowledge in social science can be inserted into machine learning through qualitative coding (Chen et al., 2018). This technique assigns inferential labels to chunks of data, enabling later model development. For example, social scientists in natural language processing (NLP) use their domain knowledge to group and structure the code system in the postprocessing step (Crowston et al., 2012). Moreover, qualitative coding is able to infer relations among latent variables. Human-understandable input and output are crucial for interpretable machine learning. If the input space contains theoretically justified components by humans, the mapping to output has logical meanings, which is much more preferred compared with pure statistical relationship (Liem et al., 2018).

In addition to social science, qualitative knowledge in natural science can be integrated into machine learning as well. For example, physical theories could guide humans to create knowledge-induced features for Higgs boson discovery (Adam-Bourdarios et al., 2015). Another strategy is to transfer language-based qualitative knowledge into numerical information. In computational molecular design (Ikebata et al., 2017), a molecule is encoded into a string including information such as element types, bond types, and branching components. The string is further processed by NLP models. The final strategy is to use experts to guide the learning process to identify a potential search direction. For instance, in cellular image annotation (Kromp et al., 2016), expert knowledge progressively improves the model through an online multi-level learning.

Concretely Formatted Knowledge

Although qualitative knowledge is relatively loose in both social and natural science, there are some formalized ways to represent it. For example, logic rules are often used to show simple relationships, such as implication ($A \rightarrow B$), equivalence ($A \leftrightarrow B$), conjunction ($A \wedge B$), disjunction ($A \vee B$), and so on. The simple binary relationship can be extended to include more entities by parenthesis association. It can be defined as a first order logic that each statement can be decomposed into a subject, a predicate, and their relationship. Logic rule–regularized machine learning models attract attention recently. For example, a "but" keyword in a sentence usually indicates that the clause after it dominates the overall sentiment. This first-order logic can be distilled into a neural network (Hu et al., 2016). In a material discovery framework called CombiFD, complex combinatorial *prioris* expressing whether the data points belong to the same cluster (Must-Link) or not (Cannot-Link) can be used as constraints in model training (Ermon et al., 2014).

Besides logic rules, invariance is another major format of qualitative knowledge which is not subject to change after transformation. An ideal strategy is to incorporate invariance in machine learning models, i.e., to build models invariant to input transformation, yet this is sometimes very difficult. Some details are shown in Section Symmetry of Convolutional Neural Networks. Besides models, one can leverage the invariant properties by preprocessing the input data. One way is to find a feature extraction method whose output is constant when the input is transformed within the symmetry space. For example, the Navier-Stokes equations obey the Galilean invariance. Consequently, seven tensors can be constructed from the velocity gradient components to form an invariant basis (Ling et al., 2016). The other way is to augment the input data based on invariance and feed the data to models (see Section Data Augmentation).

Quantitative Domain Knowledge

In scientific domains, a large amount of knowledge has been mathematically defined and expressed, which facilitates a quantitative analysis using machine learning. In the following sections, three groups of quantitative knowledge, in terms of their representation formats, are discussed: equation-based, probability-based, and graph-based knowledge.

Equation-Based Knowledge

Equality and inequality relationships can be established by algebraic and differential equations and inequations, respectively. They are the predominant knowledge format in physics, mathematics, etc. At

the same time, there is increasing amount of equation-based knowledge in chemistry, biology, engineering, and other experiment-driven disciplines. A great benefit of equations in aforementioned areas is that most variables have physical meanings understandable by humans. Even better, many of them can be measured or validated experimentally. The insertion and refinement of expert knowledge in terms of equations can be static or dynamic. Static equations often express a belief or truth that is not subject to change so that they do not capture the change of circumstance. Dynamically evolving equations, such as those used in the control area, are being used to express continuously updating processes. Equations in different categories play diverse roles in the machine learning pipeline, so equation-based knowledge can be further divided into subgroups according to their complexity.

The simplest format is a ground-truth equation, expressing consensus such as *Mass* = *Density* × *Volume*. Since it cannot be violated, this type of equation is usually treated as constraints to regularize the training process in the format of $Loss(x) = original\_Loss(x) + \sum_i \lambda_i h_i(x)$, where $h_i(x)$ is the equation-enforced regularization term and λ_i is the weight. For example, the object trajectory under gravity can be predicted by a convolutional neural network (CNN) without any labeled data, by just using a kinetic equation as the regularization term (Stewart and Ermon, 2017). The kinetic equation is easily expressed as a quadratic univariate equation of time or $h(t) = at^2 + v_0 t + h_0$. In another study, a robotic agent is designed to reach an unknown target (Ramamurthy et al., 2019). The solid body property enforces a linear relationship of segments, which serves as a regularizer in the policy architectures. The confidence of ground truth influences the degree of regularization through a soft or hard hyperparameter. For a complicated task, an expert must choose the confidence level properly.

At the second level, the equation has concrete format and constant coefficients, with single or multiple unknown variables and their derivatives. The relationship among those variables is deterministic. This means that the coefficients of these equations are state independent. Particularly, ordinary differential equation (ODE) and partial differential equation (PDE) belong to this category, which are being researched extensively within machine learning. They have the generalized form of $f\left(x_1, \ldots, x_n, \frac{\partial u}{\partial x_1}, \ldots\right) = 0$. Although only few differential equations have explicit solutions, as long as their formats can be determined by domain knowledge, machine learning can numerically solve them. This strategy inspires data-driven PDE/ODE solvers (Samaniego et al., 2020). Prior knowledge, such as periodicity, monotonicity, or smoothness of underlying physical process, is reflected in the kernel function and its hyperparameters.

In its most complicated format, equations may not fully generalize domain knowledge when the system has characteristics of high uncertainty, continuous updating, or ambiguity. The coefficients are state dependent or unknown functionals. These issues can be partially addressed by building a hybrid architecture of machine learning and PDE, in which machine learning helps predict or resolve such unknowns. For example, PDE control can be formulated as a reinforcement learning problem (Farahmand et al., 2017). The most extreme condition is that the form of coefficient/equation is unknown, but it can still be learned by machine learning purely from harnessing the experimental and simulation data. For example, governing equations expressed by parametric linear operators, such as heat conduction, can be discovered by machine learning (Raissi et al., 2017). Maximum likelihood estimation (MLE) with Gaussian process priors is used to infer the unknown coefficients from the noisy data. In another example, researchers propose to estimate nonlinear parameters in PDEs using quantum-behaved particle swarm optimization with Gaussian mutation (Tian et al., 2015). Inverse modeling can also be used to reconstruct functional terms (Parish and Duraisamy, 2016).

Probability-Based Knowledge

Knowledge in the form of probabilistic relations is another major type of quantitative knowledge. A powerful tool used in machine learning is Bayes' theorem which regulates the conditional dependence. We have prior knowledge of the relations of variables and their distributions. Given data, we could adjust some probabilities to fit observations.

Some machine learning algorithms have the intrinsic structure to capture probabilistic relations. For example, parameters of the conditional distribution in Bayesian network can be learned from data or directly from encoded knowledge (Flores et al., 2011; Masegosa and Moral, 2013). Domain knowledge can also be directly used to determine probabilities. For instance, gene relations help build optimal

Bayesian classification by mapping into a set of constraints (Boluki et al., 2017). For instance, if gene g_2 and g_3 regulate g_1 with $X_1 = 1$ when $X_2 = 1$ and $X_3 = 0$ according to the domain knowledge, the constraint can be enforced by $P(X_1 = 1|X_2 = 1, X_3 = 0) = 1$.

Graph-Based Knowledge

In both natural science and social science fields, a lot of knowledge has the "subject verb object" structure. A knowledge graph consists of entities and links among them. It can be expressed as a set of triples. The degree of their correlation can be numerically expressed and graphically visualized. Knowledge graphs are initially built by expert judgment from data. With growing size of data available, machine learning algorithms play an important role in constructing large knowledge graphs.

Google knowledge graph is an example that we access daily. It covers 570 million entities and 18 billion facts initially and keeps growing to have over 500 billion facts on ~5 billion entities (Paulheim, 2017). For example, when you search basketball in Google, highly related NBA teams and stars will appear on the right. Another famous general knowledge-formed knowledge graph is ConceptNet, which connects words and phrases of natural language with labeled edges. It can be combined with word embeddings with a better understanding of the underlying meanings (Speer et al., 2016). Specific knowledge graphs are popular in different domains. In NLP, WordNet is a semantic network to link synonyms for highlighting their meanings rather than spelling. It can enhance the performance of search applications (Fellbaum, 2012). Medical and biologic fields have, for instance, MeSH (Lowe and Barnett, 1994). It is hierarchically organized vocabulary used for indexing, cataloging, and searching of biomedical and health-related information, upon which some machine learning models are built (Abdelaziz et al., 2017; Gan et al., 2019).

Knowledge graphs and machine learning mutually benefit each other. A graph may be incomplete when there are missing entities, relations, or facts and thus needs machine learning approaches to supplement the information. For example, knowledge graph can be trained together with tasks of recommendation to connect items (Cao et al., 2019). Human-computer interaction and knowledge discovery approach (Holzinger et al., 2013) can be used to identify, extract, and formalize useful data patterns from raw medical data. Meanwhile, knowledge graph helps human to understand the related fields to promote machine learning. For instance, a better understanding of biology and neuroscience leads to advanced machine learning algorithms and neuromorphic systems (Section Human Brain and Learning). Besides, machine learning models, such as neural networks, can be built upon knowledge graph, as is further discussed in Section Design of Neuron Connections.

METHODS TO INTEGRATE HUMAN KNOWLEDGE

A complete process to design and implement a machine learning model consists of multiple steps. The first step is to formulate the appropriate tasks based on the goal. One needs to determine what the machine should learn, i.e., the inputs and outputs. After simplifying the problem by some assumptions, a model is built with unknown parameters to explain or correlate the inputs and outputs. Then, the model is initialized and trained by the collected data. After this, the machine learning model is ready for inference and evaluation. In practice, the process is not necessarily in such a chronological order but usually follows an iterative process. Some algorithms incorporate humans in the loop for feedback and evaluation. Human knowledge can be incorporated almost anywhere in this process. In the following sections, we review the methods to integrate human knowledge into machine learning. We organize them based on the sub-domains of machine learning field and group them according to their major contribution to the steps aforementioned. We should note that (1) there are numerous approaches, thus we focus on popular or emerging methods that could work efficiently across disciplines; (2) the methods are interwoven, namely, they may be used in several sub-domains and contribute to multiple steps; thus, we consider their main categories and detail them only in one place.

Task Formulation

Machine learning models, such as neural networks and support vector machines, take an array of numbers in the form of vectors or matrices as inputs and make predictions. For a given goal, there remains flexibility for humans to formulate the task, i.e., to determine the inputs and outputs of the machine learning models. Humans could combine similar tasks based on their background and shared information (Section Multitask Learning). Domain knowledge is necessary to understand and make use of the similarity of tasks. Also, we

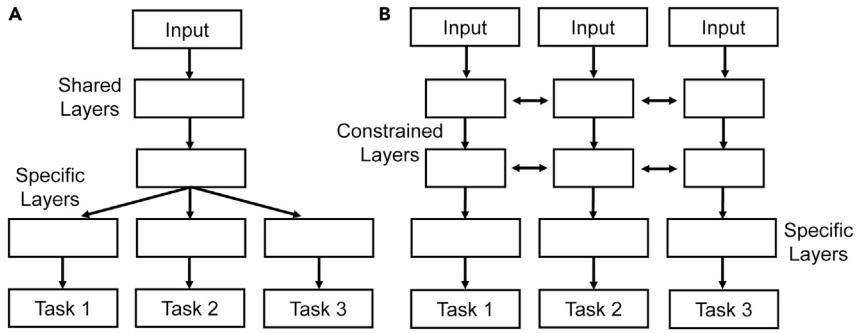

Figure 1. Illustration of Hard and Soft Parameter Sharing

(A) hard parameter sharing.

(B) soft parameter sharing.

Redrawn from (Ruder, 2017a).

need to carefully decide the inputs of machine learning models (Section Features) to best represent the essence of the tasks. We can leverage expert knowledge in the domain or some statistical tools when engineering these features.

Multitask Learning

Humans do not learn individual tasks in a linear sequence, but they learn several tasks simultaneously. This efficient behavior is replicated in machine learning with multitask learning (MTL). MTL shares knowledge between tasks so they are all learned simultaneously with higher overall performance (Ruder, 2017a). By learning the tasks simultaneously, MTL helps to determine which features are significant and which are just noise in each task (Ruder, 2017a). Human knowledge is used in MTL to determine if a group of tasks would benefit from being learned together. For example, autonomous vehicles use object recognition to drive safely and arrive at the intended destination. They must recognize pedestrians, signs, road lines, other vehicles, etc. Machine learning could be trained to recognize each object individually with supervised learning, but human knowledge tells us that the objects share an environment. The additional context increases accuracy as MTL finds a solution that fits all tasks. MTL has also been used extensively for facial landmark detection (Ehrlich et al., 2016; Ranjan et al., 2017; Trottier et al., 2017; Zhang et al., 2014) and has even contributed to medical research through drug behavior prediction (Lee and Kim, 2019; Mayr et al., 2016; Yuan et al., 2016) and through drug discovery (Ramsundar et al., 2015). Object recognition is a common use of MTL due to its proven benefits when used alongside CNNs (Girshick, 2015; Li et al., 2016).

Currently, the two main methods of task sharing are hard and soft parameter sharing as shown in Figure 1. Hard parameter sharing is where some hidden layers are shared while the output layers remain task specific. In soft parameter sharing, each task has its own model and parameters, but the parameters are encouraged to be similar through the L2 norm regularization. Currently, hard parameter sharing is more common due to being more effective in reducing overfitting. Alternative methods to hard and soft parameter sharing have been proposed, such as deep relationship networks (Long et al., 2017), cross-stitch networks (Misra et al., 2016), and sluice networks (Ruder et al., 2019). Deep relationship networks use matrices to connect the task-specific layers so they also increase the performance alongside the shared layers. Cross-stitch networks attempt to find the optimal combination of shared and specific layers. Sluice networks combine several techniques to learn which layers should be shared. These methods aim to find a general approach that works broadly so that it can be easily used for all MTL problems. However, so far, the optimal task sharing method is different for each application. This means human knowledge on the application subject and on the various task sharing methods is a necessity to find the best method for the application. It has been found that MTL is unlikely to improve performance unless the tasks and the weighting strategies are carefully chosen (Gong et al., 2019). Continued research is needed for optimal strategies to choose and balance tasks.

Another important area in MTL is to gain benefits even in the case where only one task is important. Research shows that MTL can still be used in this situation by finding an appropriate auxiliary task to support the main task (Ruder, 2017a). Similar to finding the best task sharing method, significant human knowledge is needed to find an effective auxiliary task. Another approach is to use an adversarial auxiliary task

which achieves the opposite purpose of the main task. By maximizing the loss function of the adversarial task, information can be gained for the main task. There are several other types of auxiliary tasks, but sometimes an auxiliary task is not even needed. In recent developments, MTL principles are utilized even in single task settings. Pseudo-task augmentation is where a single task is being solved; however, multiple decoders are used to solve the task in different ways (Meyerson and Miikkulainen, 2018). Each solving method is treated as a different task and implemented into MTL. This allows a task to be solved optimally as each method of solving the task learns from the other methods.

Features

"Features" in machine learning refer to variables that represent the property or characteristic of the observations. They could be statistics (e.g., mean, deviation), semantic attributes (e.g., color, shape), transformation of data (e.g, power, logarithm), or just part of raw data. "Feature engineering" is the process to determine and obtain input features to optimize machine learning performance. In this section, we will discuss the approaches in this process.

Feature engineering, especially feature creation, heavily relies on human knowledge and experience in the areas. For instance, in a credit card fraud detection task, there are many items associated with each transaction, such as transaction amount, merchant ID, and card type. A simple model treats each transaction independently and classifies the transactions by eliminating unimportant items (Brause et al., 1999). Later, people realize that costumer spending behaviors matter as well. Then, their indicators, such as transactions during the last give number of hours and countries, are aggregated (Whitrow et al., 2009). This methodology is further amended by capturing transaction time and its periodic property (Bahnsen et al., 2016). We could see from this example that it is an iterative process that interplays with feature engineering and model evaluation (Brownlee, 2014). Although the guidelines vary with specific areas, a rule of thumb is to find the best representation of the sample data to learn a solution.

In addition to domain knowledge, there are some statistical metrics in feature selection widely used in different areas. There are mainly two types of ideas (Ghojogh et al., 2019). One is called filter methods, which rank the features by their relevance (correlation of features with targets) or redundancy (whether the features share redundant information) and remove some by setting a threshold. This type includes linear correlation coefficient, mutual information, consistency-based filter (Dash and Liu, 2003), and many others. The other type is called wrapper methods. These methods train and test the model during searching. They look for the subset of features that correspond to the optimal model performance. Since the number of combinations grow exponentially with the number of features, it is a non-deterministic polynomial-time (NP) hard problem to find the optimal subset. One searching strategy, sequential selection methods, is to access the models sequentially (Aha and Bankert, 1996). The other strategy is to implement metaheuristic algorithms such as the binary dragonfly algorithm (Mafarja et al., 2017), genetic algorithm (Frohlich et al., 2003), and binary bat algorithm (Nakamura et al., 2012b). Wrapper methods can be applied to simple models instead of the original ones to reduce computation. For instance, Boruta (Kursa and Rudnicki, 2010) uses a wrapper method on random forests, which in essence can be regarded as a filter method. In addition to these two types, there exist other techniques such as embedded methods, clustering techniques, and semi-supervised learning (Chandrashekar and Sahin, 2014).

"Feature learning", also called "representation learning", is a set of techniques that allow machine to discover or extract features automatically. Since the raw data may contain redundant information, we normally want to extract features with lower dimension, i.e., to find a mapping $\mathbb{R}^d \rightarrow \mathbb{R}^p$ where $p \leq d$ and usually $p \ll d$. Thus, these methods are sometimes referred to as dimension reduction. Traditional ways are statistical methods to extract features. Unsupervised (unlabeled data) methods include principal component analysis (Wold et al., 1987), maximum variance unfolding (Weinberger and Saul, 2006), Laplacian eigenmap (Belkin and Niyogi, 2003; Chen et al., 2010), and t-distributed stochastic neighbor embedding (t-SNE) (Maaten and Hinton, 2008). Supervised (labeled data) methods include Fisher linear discriminant analysis (Fisher, 1936), its variant Kernel Fisher linear discriminant analysis (Mika et al., 1999), partial least squares regression (Dhanjal et al., 2008), and many approaches on supervised principal component analysis (Bair and Tibshirani, 2004; Barshan et al., 2011; Daniušis et al., 2016). Recent works are mostly based on neural networks, such as autoencoders (Baldi, 2012), CNNs, deep Boltzman machines (Salakhutdinov and Hinton, 2009). Even though these methods extract features automatically, prior knowledge can still be incorporated. For instance, to capture the features of videos with slow moving objects, we can represent the

objects by a group of numbers (such as their positions in space and the pose parameters) rather than by a single scalar, and these groups tend to move together (Bengio et al., 2013; Hinton et al., 2011). Or we can deal with the moving parts and the static background separately (Li et al., 2018). Another example, which applies the principal of Multitask Learning, is to train an image encoder and text encoder simultaneously and correlate their extracted features (Reed et al., 2016). Other strategies to integrate knowledge include data manipulation and neural network design, as will be discussed in the following sections.

Model Assumptions

Machine learning models are built upon assumptions or hypotheses. Since "hypothesis" in machine learning field commonly refers to model candidates, we use the word "assumption" to denote explicit or implicit choices, presumed patterns, and other specifications on which models are based for simplification or reification. The assumptions could be probabilistic or deterministic. We first introduce probabilistic assumptions in Sections Preliminaries of Probabilistic Machine Learning, Variable Relation and Distribution, and then briefly discuss Deterministic Assumptions.

Preliminaries of Probabilistic Machine Learning

Mathematically, what are we looking for when we train a machine learning model? From the perspective of probability, two explanations have been proposed as shown in Equations (1) and (2) (Murphy, 2012). One is called MLE (maximum likelihood estimation) which means that the model maximizes the likelihood of the training data set,

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}}\, p(\mathcal{D}|\theta) \qquad \text{(Equation 1)}$$

where $\mathcal{D}$ is the training data set and $p(\mathcal{D}|\theta)$ is the probability of data provided by the machine learning model whose parameter is θ. The machine learning model estimates the probability of observation, while the training process tries to find the parameter which best accords with the observation. The specific form of $\mathcal{D}$ depends on models. For example, in supervised learning, we can rewrite the form as $p(Y|X,\theta)$, i.e., the probability of label Y given input X and model parameters θ, and we regard the label with the highest probability as the model's prediction.

The other strategy, maximum a posteriori (MAP), means to maximize the posterior probability of the model parameter,

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\, p(\theta|\mathcal{D}) = \underset{\theta}{\operatorname{argmax}}\, \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \underset{\theta}{\operatorname{argmax}}\, p(\mathcal{D}|\theta)p(\theta) \qquad \text{(Equation 2)}$$

MAP takes into account $p(\theta)$, the *prior* distribution of θ. From the equations above, we can observe that MLE is a special case of MAP where $p(\theta)$ is uniform.

Variable Relation

A variable could be an instance (data point), a feature, or a state of the system. An assumption on variable relation used in almost all models is the independence of data instances; therefore, the probability of data sets is equal to the product of instance probabilities. For instance, in unsupervised learning we have $p(X|\theta) = \prod_i p(x^{(i)}|\theta)$.

When the variables are related, we could simplify the likelihood function by assuming partial independence. For instance, in image generation models, pixels are correlated to each other; thus, an image x is associated with all pixels x_i ($i = 1, 2, \ldots$). Pixel RNN (Oord et al., 2016a) assumes that pixel x_i is only correlated to its previous pixels and independent of those afterward:

$$p(x) = p(x_1, x_2, \ldots, x_N) \approx \prod_{i=1}^{N} p(x_i|x_1, x_2, \ldots, x_{i-1}) \qquad \text{(Equation 3)}$$

Under this assumption, pixels are generated sequentially by an RNN. As a simpler implementation, we can pay more attention to the local information and calculate a pixel value by the previous ones close to it (Oord et al., 2016b).

In the stochastic process of time series, Equation (3) is commonly used, and x_i denotes the variable value at time i. In this context, the approximation is more of knowledge than an assumption since it is hard to imagine that the future would impact the past. It can be simplified further by assuming that future probabilities

are determined by their most recent values $p(x_i|x_1, x_2, ..., x_{i-1}) \approx p(x_i|x_{i-1})$ and then the variables x_i, called *states*, form a Markov chain. A more flexible model, hidden Markov model (HMM), treats the states as hidden variables and the observations have a conditional probability distribution given the states. HMM is widely used in stock market, data analysis, speech recognition, and so forth (Mor et al., 2020). Such Markov property is also presumed in reinforcement learning (Section Reinforcement Learning), where the next state is determined by the current state and action.

In addition to sequential dependence, the relation between variables can be formulated as a directed acyclic graph or a Bayesian network in which nodes represent the variables and edges represent the probabilistic dependencies. Learning the optional structure of a Bayesian network is NP-hard problem, which means that it requires a huge amount of computation and may not converge to global optima. Prior knowledge of the variables can be incorporated by their dependencies, such as the existence or absence of an edge and even the probability distribution (Su et al., 2014; Xu et al., 2015). The results of learned networks get improved while computational cost is reduced.

Distribution

The distribution of variables is unknown and has to be assumed or approximated by sampling. A very popular and fundamental distribution is the Gaussian distribution, a.k.a. normal distribution. Its popularity is not groundless; instead, it is based on the fact that the mean of a large number of independent random variables, regardless of their own distributions, tends toward Gaussian distribution (central limit theorem). The real-world data are composed of many underlying factors, so the aggregate variables tend to have a Gaussian distribution in nature. Some models are named after it, such as Gaussian process (Ebden, 2015) and Gaussian mixture model (Shental et al., 2004). Independent component analysis (ICA) decomposes the observed data into several underlying components and sources (Hyvärinen, 2013). ICA assumes that the original components are non-Gaussian. In linear regression models, the least square method can be derived from Equation (1) while assuming the output Y has a Gaussian distribution with a mean of $\theta^T X$, namely $p(\mathcal{D}|\theta) = p(Y|X, \theta) = \mathcal{N}(\theta^T X, \sigma^2)$ where σ is the standard deviation. Further, by assuming the prior $p(\theta)$ to be Gaussian with zero mean, regularized linear regression can be derived from Equation (2), whose loss function is penalized by the sum of the squares of the terms in θ.

Besides Gaussian, other types of distribution assumptions are applied as well. Student-t process is used in regression to model the priors of functions (Shah et al., 2014). In many Bayesian models, priors must be given through either the training data or manual adjustment. Inaccurate priors would cause systemic errors of the models. For instance, in spam email classification by naive Bayesian classifiers, if assuming uniform distribution of spam and non-spam (50/50), then the classifier is prone to report a non-spam email as spam when applying it to real life where the ratio of spam emails is much smaller, say 2%. Conversely, assuming a 2% spam ratio, the classifier would miss spam emails when applying it to email accounts full of spam emails. Although methods have been proposed to alleviate this issue (Frank and Bouckaert, 2006; Rennie et al., 2003), classifiers would benefit from appropriate priors. Such training-testing mismatch problem also occurs in other machine learning models. To the best of our knowledge, there is no perfect solution to this problem; thus, the most efficient way is to design a distribution of training data close to test scenarios which requires knowledge from experts.

Deterministic Assumptions

Deterministic assumptions describe the properties and relations of objects. Some deterministic assumptions can also be expressed in a probabilistic way, such as variable independence. In addition to these, many are encoded in the "hypothesis space": for an unknown function $f : X \rightarrow Y$, we use a machine learning model to approximate the target function. Hypothesis space is the set of all the possible functions. This is the set from which the algorithm determines the model which best describes the target function according to the data. For instance, in artificial neural networks, the configuration of the networks, e.g., the number of layers, activation functions, and hyperparameters, is determined priorly. Then, all combinations of weights span the hypothesis space. Through training, the optimal combination of weights is calculated. The design of networks can be integrated with human knowledge, which is elaborated in the section below.

Network Architecture

Artificial neural network has been proved to be a very powerful tool to fit data. It is extensively used in many machine learning models especially after the emergence of deep learning (LeCun et al., 2015). Although

ideally neural networks could adapt to all functions, adding components more specific to the domains can boost the performance. As mentioned in Section Deterministic Assumptions, the architecture of network regulates the hypothesis space. Using a specialized network reduces the size of hypothesis space and thus generalizes better with less parameters than universal networks. Therefore, we want to devise networks targeting at data and tasks.

There have been network structures proposed for specific tasks. For instance, RNNs are applied to temporal data, such as language, speech, and time series; capsule neural networks are proposed to capture the pose and spatial relation of objects in images (Hinton et al., 2011). In the following contents, we elaborate how symmetry is used in CNNs and how to embed different knowledge via customizing the neuron connections.

Symmetry of Convolutional Neural Networks

Symmetry, in a broad sense, denotes the property of an object which stays the same after a transformation. For instance, the label and features of a dog picture remain after rotation. In CNNs, symmetry is implemented by two concepts, "invariance" and "equivariance". Invariance means the output stays the same when the input changes, i.e., $f(Tx) = f(x)$ where x denotes the input, T denotes the transformation, and f denotes the feature mapping (e.g. convolution calculation). Equivariance means the output preserves the change of the input, i.e., $f(Tx) = T'[f(x)]$ where T' is another transformation that could equal T. CNNs are powerful models for sensory data and especially images. A typical CNN for image classification is composed of mostly convolution layers possibly followed by pooling layers and fully connected ones for the last few layers. Convolution layers use filters to slide through images and calculate inner products with pixels to extract features to analyze each part of images. Such weight sharing characteristic not only greatly reduces the number of parameters but also provides them with inherent translation equivariance: the convolution output of a shifted image is the same as the shifted output of the original image. Basically, the CNN relies on convolution layers for equivariance and fully connected layers for invariance.

The inherent translation equivariance is limited: there are reports showing that the confidence of correct labels would decrease dramatically even with shift unnoticeable by human eyes. It is ascribed to the aliasing caused by down-sampling (stride) commonly used on convolution layers, and a simple fix is to add a blur filter before down-sampling layers (Zhang, 2019). Other symmetry groups are also considered, such as rotation, mirror, and scale. The principle of most works is to manipulate filters since they compute faster than data transformation. A simple idea is to use symmetric filters such as circular harmonics (Worrall et al., 2017) for rotation or log-radial harmonics (Ghosh and Gupta, 2019) for scale. Such equivariance or invariance is local, that is, only each pixel-filter operation has the symmetry property while the whole layer output, composed of multiple operations, does not. A global way is to traverse the symmetry space and represent with exemplary points (control points) (Cohen and Welling, 2016). For instance, in the previous dog example, we could rotate the filter by 90, 180, and 270° to calculate the corresponding feature maps. Thus, we can approximate the equivariance of rotation. Kernel convolution can be used to control the extent of symmetry, e.g. to distinguish "6" and "9" (Gens and Domingos, 2014). Overall, although invariant layers (Kanazawa et al., 2014) can also be constructed to incorporate symmetry, equivariant feature extraction as intermediate layers is preferred in order to preserve the relative pose of local features for further layers.

Design of Neuron Connections

By utilizing the knowledge of invariance and equivariance in the transformation, the CNN preforms well in image classification especially with some designed filters. From another perspective, we can also say that the CNN includes the knowledge of graphs. Imagine a pixel in a graph, which is connected to the other pixels around it. By pooling and weights sharing, the CNN generalizes the information of the pixel and its neighbors. This kind of knowledge is also applied in a graph neural network (GNN) (Gori et al., 2005; Wu et al., 2020) where the node's state is updated based on embedded neighborhood information. In the GNN, the neighbors are not necessarily the surrounding pixels but can be defined by designers. Thus, the GNN is able to represent the network nodes as low-dimensional vectors and preserve both the network topology structure and the node content information. Furthermore, the GNN can learn more informative relationship through differential pooling (Ying et al., 2018) and the variational evaluation-maximization algorithm (Qu et al., 2019).

Not only is the knowledge of general connections between nodes used in network design but also the specific relational knowledge in connections is beneficial. Encoding the logic graph (Fu, 1993) and hierarchy

relationships (Deng et al., 2014) into the architecture, such as "$A \wedge B \rightarrow C$" and "$a \in A$", is also one way to build neural networks with Graph-Based Knowledge. With these encoding methods, some of the rules are learned from data and part of them are enforced by human knowledge. This idea is reflected in the cooperation of symbolic reasoning and deep learning, which is getting increasingly popular recently (Garnelo and Shanahan, 2019; Mao et al., 2019; Segler et al., 2018). Symbolic reasoning or symbolic artificial intelligence (AI) (Flasiński, 2016; Hoehndorf and Queralt-Rosinach, 2017) is a good example of purely utilizing graph-based knowledge where all the rules are applied by humans and the freedom of learning is limited, while in deep learning, the rules are automatically learned from data. Building symbolic concept into neural networks helps increase the network's interpreting ability and endow the networks with more possibilities, allowing more interactions between labels, for instance.

Aside from the knowledge of graphs, Equation-Based Knowledge can also be united with deep learning. For example, a general data-driven model can be added to a first-principle model (Zhou et al., 2017). The add-on neural networks will learn the complex dynamics which might be impossible to identify with pure physical models, e.g. learning the close-to-ground aerodynamics in the drone landing case (Shi et al., 2019). Another example in utilizing equation-based knowledge is using optimization equations in the layers through OptNet (Amos and Kolter, 2017), a network architecture which allows learning the necessary hard constraints. In these layers, the outputs are not simply linear combinations plus nonlinear activation functions but solutions (obtained by applying Karush-Kuhn-Tucker conditions) to constrained optimization problems based on previous layers.

In speech recognition, the words in a sentence are related, and the beginning of the sentence may have a huge impact on interpretation. This induces delays and requires accumulating information over time. Also, in many dynamical systems, especially those involving human reaction time, the effects of delays are important. The knowledge of delay is then introduced in the design of neural networks. For example, time-delay neural networks (Waibel et al., 1989) take information from a fixed number of delayed inputs and thus can represent the relations between sequential events. Neural networks with trainable delay (Ji et al., 2020) utilize the knowledge of delay's existence and learn the delay values from data. RNN (Graves et al., 2013; Zhang et al., 2019) is a network architecture in which neurons feedback in a similar manner to dynamical systems, where the subsequent state depends on the previous state. Through inferring the latent representations of states instead of giving label to each state, the RNN can be trained directly on text transcripts of dialogs. However, these end-to-end methods often lack constraints and domain knowledge which may lead to meaningless answers or unsafe actions. For instance, if a banking dialog system does not require the username and password before providing account information, personal accounts could be accessed by anyone. A hybrid code network (HCN) (Williams et al., 2017) is proposed to address this concern. The HCN includes four components: entity extraction module, RNN, domain-specific software, and action templates. The RNN and domain-specific software maintain the states, and the action templates can be a textual communication or an API call. This general network allows experts to express specific knowledge, achieves the same performance with less data, and retains the benefits of end-to-end training.

Data Augmentation

Machine learning, especially deep learning, is hungry for data. The problems with insufficient data include overfitting where the machine generalizes poorly and class imbalance where the machine does not learn what we want because real-world data sets only contain a small percentage of "useful" examples (Salamon and Bello, 2017). These problems can be addressed by data augmentation, a class of techniques to artificially increase the amount of data with almost no cost. The basic approaches are transforming, synthesizing, and generating data. From the perspective of knowledge, it teaches machine invariance or incorporates knowledge-based models. Some papers do not consider simulation as data augmentation, but we will discuss it here since in essence these techniques are all leveraging human knowledge to generate data. Data augmentation needs more computation than explicit ways (e.g. Section Symmetry of Convolutional Neural Networks), but it is widely used in many areas such as Image, Audio, time series (Wen et al., 2020), and NLP (Fadaee et al., 2017; Ma, 2019) owing to its flexibility, simplicity, and effectiveness. It is even mandatory in unsupervised representation learning (Chen et al., 2020; He et al., 2020).

Image

Some representative data augmentation techniques for images are illustrated in Figure 2. A fundamental approach is to apply affine transformation to geometries, i.e., cropping, rotating, scaling, shearing, and

translating. Noise and blur filters can be injected for better robustness. Elastic distortions (Simard et al., 2003), designed for hand-written character recognition, are generated using a random displacement field in image space to mimic uncontrolled oscillations of muscles (Wong et al., 2016). Random erasing (Zhong et al., 2020) is analogous to dropout except that it is applied to input data instead of network architecture. Kernel filters are used to generate blurred or sharpened images. Some kernels (Kang et al., 2017) can swap the rows and columns in the windows. This idea of "swapping" is similar to another approach, mixing images. There are two ways to mix images, one is cropping and merging different parts of images (Summers and Dinneen, 2019), the other is overlapping images and averaging their pixel values (Inoue, 2018). They have both been demonstrated to improve performance, though the latter is counterintuitive.

Another perspective is to change the input data in color space (Shorten and Khoshgoftaar, 2019). Images are typically encoded by RGB (red, green, blue) color space, i.e. represented by three channels (matrices) indicating red, green, and blue. These values are dependent upon brightness and lightning conditions. Therefore, color space transformation, also called photometric transformation, can be applied. A quick transformation is to increase or decrease the pixel values of one or three channels by a constant. Other methods are, for instance, setting thresholds of color values and applying filters to change the color space characteristics. Besides RGB, there are other color spaces such as CMY (cyan, magenta, yellow), HSV (hue, saturation, value which denotes intensity), and CIELab. The performance varies with color spaces. For example, a study tested four color spaces in a segmentation task and found that CMY outperformed the others (Jurio et al., 2010). It is worthy to note that human judgment is important in the freedom of color transformation since some tasks are sensitive to colors, e.g. distinguishing between water and blood. In contrast to augmentation, another direction to tackle color variance is to standardize the color space such as adjusting the white balance (Afifi and Brown, 2019).

Deep learning can be used to generate data for augmentation. One way is adversarial training which consists of two or more networks with contrasting objectives. Adversarial attacking uses networks to learn augmentations to images that result in misclassifications of their rival classification networks (Goodfellow et al., 2014; Su et al., 2019). Another method is generative models, such as generative adversarial networks and variational auto-encoders. These models can generate images to increase the amount of data (Lin et al., 2018). Style Transfer (Gatys et al., 2015), best known for its artistic applications, serves as a great tool for data augmentation (Jackson et al., 2019). In addition to images in the input space, data augmentation can also be applied to the feature space, i.e., the intermediate layers of neural networks (Chawla et al., 2002; DeVries and Taylor, 2017).

Audio

There are many types of audio, such as music and speech, and accordingly many types of tasks. Although augmentation method varies with audio types and tasks, the principles are universal.

Tuning is frequently used by music lovers and professionals to play or post-process music. There is a lot of mature software for it. They can stretch time to shift pitch or change the speed without pitch shifting. More advanced tuning involves reverberation (sound reflection in a small space), echo, saturation (non-linear distortion caused by overloading), gain (signal amplitude), equalization (adjusting balance of different frequency components), compression, etc. These methods can all be used in audio data augmentation (Mignot and Peeters, 2019; Ramires and Serra, 2019). Unfavorable effects in tuning, such as noise injection and cropping, are used as well in audio data augmentation.

An interesting perspective is to convert audio to images on which augmentations are based. Audio waveforms are converted to spectrograms which represent the intensity of a given frequency (vertical axis) at a given time (horizontal axis). Then, we can modify the "image" by distorting in the horizontal direction, blocking several rows, or blocking several columns (Park, 2019). These augmentations help the networks to be robust against, respectively, time direction deformations, partial loss of frequency channels, or partial loss of temporal segments of the input audio. This idea was extended by introducing more policies such as vertical direction distortion (frequency warping), time length control, and loudness control (Hwang et al., 2020). Other techniques used in image augmentation, e.g., rotation and mixture, are attempted as well (Nanni et al., 2020).

Simulation

As humans create faster and more accurate knowledge-based models to simulate the world, using simulations to acquire a large amount of data becomes an increasingly efficient method for machine learning. The primary

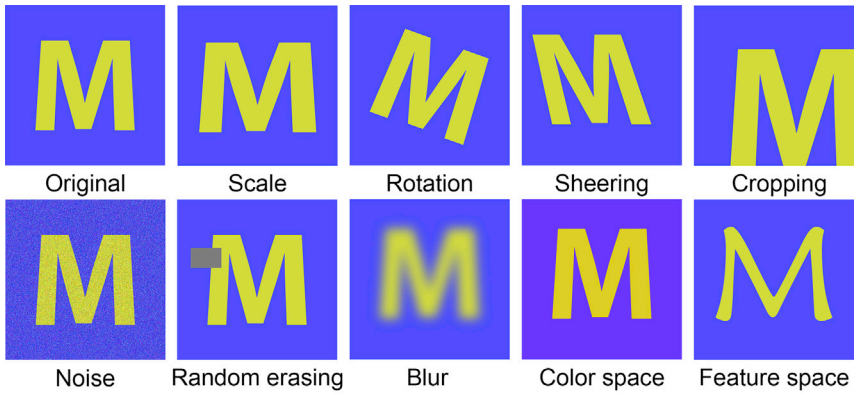

Figure 2. Illustration of Image Augmentation Techniques

advantage of simulations is the ability to gather a large amount of data when doing so experimentally would be costly, time consuming, or even dangerous (Ruder, 2017b; Ruiz et al., 2018). In some cases, acquiring real-world data may even be impossible without already having some training through simulations.

One application that conveys the important role of human knowledge in simulation data is computer vision. Humans can use their visual knowledge to develop powerful visual simulations to train computers. Autonomous vehicles, for instance, can be trained by simulated scenarios. They can learn basic skills before running on the road and can also be exposed to dangerous scenes that are rare naturally. Open-source simulation environments and video games such as Grand Theft Auto V (Martinez et al., 2017) can be used to reduce the time and money required to build simulations. Besides autonomous vehicles, simulation data have been used for computer vision in unique works such as cardiac resynchronization therapy (Giffard-Roisin et al., 2018), injection molding (Tercan et al., 2018), and computerized tomography (CT) scans (Holmes et al., 2019). Each of these applications requires thorough human knowledge of the subject. Lastly, robotics is a field where simulation data are expected to play a significant role in future innovation. Training robotics in the real world is too expensive, and the equipment may be damaged (Shorten and Khoshgoftaar, 2019). By incorporating human models, simulation data can allow robotics to be trained safely and efficiently.

Improvement through future research will accelerate the adoption of this technique for more tasks. Human experience and knowledge-based models will make simulations in general more realistic and efficient. Therefore, simulation data will become even more advantageous for training. At the same time, data-driven research aims to find optimal simulations that provide the most beneficial data for training the real-world machine. For instance, reinforcement learning is used to quickly discover and converge to simulation parameters that provide the data which maximizes the accuracy of the real-world machine being trained (Ruiz et al., 2018). While data-driven methods will reduce the human knowledge necessary for controlling the simulation, they will not replace the necessity of human knowledge for developing the simulation in the first place.

Feedback and Interaction

As humans, we gain knowledge mostly from interactions with the environment. In some algorithms, the machine is designed to interact with humans. Including human-in-loop (Holzinger, 2016; Holzinger et al., 2019) can help interpret the data, promote the efficiency of learning, and enhance the performance.

A typical method that demonstrates how machines can interact with the environment, including humans and preset rules, is discussed in Section Reinforcement Learning. Knowledge can also be injected through the rewards as well. In Section Active Learning, machines may ask humans for data labeling or distribution. In Section Interactive Visual Analytics, machines interact with humans by bringing new knowledge through visualization while seeking manual tuning.

Reinforcement Learning

Reinforcement learning is a goal-directed algorithm (Sutton and Barto, 2018) which learns the optimal solution to a problem by maximizing the benefit obtained in interactions with the environment.

In reinforcement learning, the component which makes decisions is called "agent" and everything outside and influenced by the agent is "environment". At time t, the agent has some information about the environment which can be denoted as a "state" S_t. According to the information, it will take an *action* A_t under a *policy* $\pi_t(a|s)$, which is the probability of choosing $A_t = a$ when $S_t = s$. This action will lead to the next state S_{t+1} and a numerical *reward* R_{t+1}. The transition between two states is given by interactions with the environment and is described by the environment model. If the probability of transition between states is given, we can solve the problem by applying model-based methods. When the model is unknown, which is usually the case in real life, we can also learn the model from the interactions and use the learned model to simulate the environmental behaviors if interactions are expensive.

The agent learns the optimal policy by maximizing the accumulated reward G_t (usually in episodic problems and is called return) or average reward (in continuing problems). In general, the collected rewards can be evaluated by state value $V_\pi(s) := \mathbb{E}_\pi[G_t|S_t = s]$ or action value $Q_\pi(s,a) := \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$. Based on different policy evaluation approaches, the methods used in reinforcement learning can be categorized into two types. One type is to evaluate the policy by the value function $V_\pi(s)$ or $Q_\pi(s,a)$. The value function is estimated through table (tabular methods) or approximator (function approximation methods). In the tabular methods, the exact values of those states or state-action pairs are stored in a table; thus, the tabular methods are usually limited by computation and memory (Kok and Vlassis, 2004). Function approximation (Xu et al., 2014) provides a way to bypass the computation and memory limit in high-dimensional problems. The states and actions are generalized into different features, then the value functions become functions of those features, and the weights in the function are learned through interactions. The other type is to evaluate the policy directly by an approximator. Apart from learning the value functions, an alternative approach in reinforcement learning is to express the policy with its own approximation, which is independent of the value function. These kinds of methods are called policy gradient methods, including actor-critic methods which learn approximations to both policy and value functions (Silver et al., 2014; Sutton et al., 2000). With these methods, the agent learns the policy directly.

Feedback from the human or the environment as a reward in reinforcement learning is essential since it is a goal-directed learning algorithm. There are many tasks for which human experience remains useful, and for those tasks, it would be efficient and preferable to obtain the knowledge from humans directly and quickly. Humans can participate in the training process of reinforcement learning in two ways, one is to indirectly shape the policy by constructing the reward function, while the other is to directly intervene with the policy during learning. In the former way, when the goal of some tasks is based on human satisfaction, we need humans to give the reward signal and ensure that the agent fulfills the goal as we expect (Knox and Stone, 2008). Including human rewards, the policy is pushed indirectly toward the optimal one under the human's definition, and the learning process is sped up (Loftin et al., 2016). Aside from giving reward manually after each action, human knowledge can also be used to design the reward function, e.g. give more positive weights to those important indicators in multi-goal problems (Hu et al., 2019). A recent work summarizes how to inject human knowledge into a tabular method with reward shaping (Rosenfeld et al., 2018). In the other way, guidelines from humans directly exist in the policy. Human feedback can modify the exploration policy of the agent and participate in the action selection mechanism (Knox and Stone, 2010, 2012). The feedback on policy can be not only a numerical number but also a label on the optimal actions (Griffith et al., 2013). By adding the label, human feedback changes the policy directly instead of influencing the policy through rewards. Recent works show that human feedback also depends on the agent's current policy which enables useful training strategies (MacGlashan et al., 2017), and involving humans in the loop of reinforcement learning gives improvement in learning performance (Lin et al., 2017).

Active Learning

Active learning, by selecting partial data to be annotated, aims to resolve the challenge that labeled data are more difficult to obtain than unlabeled data. During the training process, the learner poses "queries" such as some unlabeled instances to be labeled by an "oracle" such as a human. An example of active learning algorithm for classification is shown in Figure 3. Initially, we have some labeled data and unlabeled data. After training a model based on the labeled data, we can search for the most informative unlabeled data and query the oracle to obtain its label. Eventually, we can have an excellent classifier with only few additional labeled data. In this scenario, the learner makes a decision on the query after evaluating all the instances; thus, it is called "pool-based sampling". Other scenarios include "stream-based selective sampling" (each unlabeled data point is examined one at a time with the learner evaluating the

$\mathcal{U} = \{x^{(1)}, x^{(2)}, x^{(3)}, ...\}$ (Set of unlabeled data)
$\mathcal{L} = \{\langle x^{(1)}, y^{(1)}\rangle, \langle x^{(2)}, y^{(2)}\rangle, \langle x^{(3)}, y^{(3)}\rangle, ...\}$ (Set of labeled data)
for $t = 1, 2, ...$ **do**
  Train a classifier f from $\mathcal{L}$
  Select the most informative instance $x^* \in \mathcal{U}$ according to f
  Query the oracle to obtain label y^*
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{\langle x^*, y^*\rangle\}$
  $\mathcal{U} \leftarrow \mathcal{U} \backslash \{x^*\}$
end for

Figure 3. Pseudocode of an Active Learning Example
Rephrased from (Settles, 2012)

informativeness and deciding whether to query or discard) and "query synthesis" (the learner synthesizes or creates its own instance).

There are many ways to define how informative a data point is, namely, "query strategies" vary. As illustrated in Figure 4, effective query strategies help the trained model outperform the one trained by random sampling. Therefore, it is a major issue in active learning to apply optimal query strategy, which has the following categories (Settles, 2012): (1) uncertainty sampling, which measures the uncertainty of the model's prediction on data points; (2) query by disagreement, which trains different models and checks their differences; (3) error and variance reduction, which directly looks into the generalization error of the learner. Besides, there are many other variants, such as density or diversity methods (Settles and Craven, 2008; Yang et al., 2015), which consider the repressiveness (reflection on input distribution) of instances in uncertainty sampling, clustering-based approaches (Dasgupta and Hsu, 2008; Nguyen and Smeulders, 2004; Saito et al., 2015) which cluster unlabeled data and query the most representative instances of those clusters, and min-max framework (Hoi et al., 2009; Huang et al., 2010) which minimizes the maximum possible classification loss. More versatile methods include combining multiple criteria (Du et al., 2015; Wang et al., 2016; Yang and Loog, 2018), choosing strategies automatically (Baram et al., 2004; Ebert et al., 2012), and training models to control active learning (Bachman et al., 2018; Konyushkova et al., 2017; Pang et al., 2018).

In addition to asking the oracle to label instances, queries may seek for more advanced domain knowledge. A simple idea is to solicit information about features. For instance, besides instance-label queries, a text classifier (Raghavan et al., 2006) may query the relevance between features (words) and classes, e.g. "is 'puck' discriminate to determine whether documents are about basketball or hockey?" Then, the vectors of instances are scaled to reflect the relative importance of features. Another way is to set constraints based on high-level features (Druck et al., 2009; Small et al., 2011). The oracle may be queried on possibilities, e.g. "what is the percentage of hockey documents when the word 'puck' appears?" The learning algorithm then tries to adjust the model to match the label distributions over the unlabeled pool. Other methods to incorporate features include adjusting priors in naive Bayes models (Settles, 2011), mixing models induced from rules and data (Attenberg et al., 2010), and label propagation in graph-based learning algorithms (Sindhwani et al., 2009). Humans sometimes do a poor job in answering such questions, but it is found that specifying many imprecise features may result in better models than fewer more precise features (Mann and McCallum, 2010).

Although most of active learning work is on classification, the principals also apply to regression (Burbidge et al., 2007; Willett et al., 2006). Recent work focuses on leveraging other concepts for larger data sets (e.g., image data), such as deep learning (Gal et al., 2017), reinforcement learning (Liu et al., 2019), and adversary networks (Sinha et al., 2019).

Interactive Visual Analytics

Visual analytics (VA) is a field where information visualization helps people understand the data and concepts in order to make better decisions. One core in VA is dimension reduction which can be well addressed through machine learning (Sacha et al., 2016). Applying the interactive VA, which allows users to give feedback in the modeling-visualizing loop, will make machine learning tools more approachable in model understanding, steering, and debugging (Choo and Liu, 2018).

In the combination of machine learning and VA, human knowledge plays an indispensable role as interactions and feedback to the system (Choo et al., 2010). Interactions and feedback can happen either in the visualization part or the machine learning part. In the former part, visualization systems satisfy users'

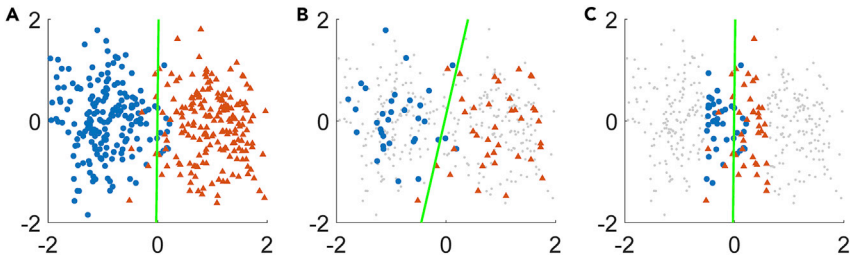

Figure 4. An Illustration of Active Learning: Choosing Data to Inquire for Better Estimation When Labeled Data Are Not Sufficient

Data shown are randomly generated from two Gaussian distributions with different means. Drawn based on the concept in (Settles, 2012).

(A) Correct labels of the binary classification problem. The line denotes the decision boundary.

(B) A model trained by random queries.

(C) A model trained by active queries.

requirements through interacting with them and assisting users in having a better understanding of the data analyzed by some machine learning methods. For instance, principle component analysis (PCA) is a powerful machine learning method which transforms the data from the input space to the eigenspace and reduces the dimension of the data by choosing the most representative components. However, for many users, PCA works as a "black box" and it is difficult to decipher the relationships in the eigenspace. Interactive PCA (Jeong et al., 2009) provides an opportunity for the users to give feedback on system visualization, and these interactions are reflected immediately in other views so that the user can identify the dimension in both the eigenspace and the original input space. In the machine learning part, interactions and feedback from humans help machine learning methods to generate more satisfying and explainable results and make the results easier to visualize as well. A good example of utilizing human interactions in machine learning methods is the application in classification (Fails and Olsen, 2003; Ware et al., 2001). These interactive classifiers allow users to view, classify, and correct the classifications during training. Readers can refer to a recent comment (Holzinger et al., 2018) on explainable AI.

The mutual interaction between human knowledge and the visualization or the model is an iterative process. A better visualization leads the users to learn more practical information. After the users gain some knowledge regarding the model and data, they can utilize the knowledge to further improve the model and even the learning algorithm (Hohman et al., 2018). This iterative process has a steering effect on the model, which can be viewed as the parameter evolution in dynamical systems shown in Equation (4) (Dıaz et al., 2016; Endert et al., 2017):

$$\dot{y} = f(y,u),\ v = g(y), \qquad \text{(Equation 4)}$$

where y is the model under the machine learning algorithm, v is the visualization of that model, and $u = \{x, w\}$ is the input including the new input data x and users' feedback w. The feedback w is based on users' knowledge as well as the visualization v. Training of the model is complete when the dynamical system settles down at a certain equilibrium, y^*.

Some progress has been made in this interdisciplinary area to help machine learning become more accessible. Semantic interaction (Endert et al., 2012) is an approach that enables co-reasoning between the human and the analytic models used for visualization without directly controlling the models. Users can manipulate the data during visualization, and the model is steered by their actions. In this case, interactions happen with the help of visualization and affect both model and visual results. Interactive visual tools can also be built for understanding and debugging current machine learning models (Bau et al., 2019; Karpathy et al., 2015; Zeiler and Fergus, 2014). The language model visual inspector system (Rong and Adar, 2016) is able to explore the word embedding models. It allows the users to track how the hidden layers are changing and inspect the pairs of words. The reinforcement learning VA systems, DQNViz (Wang et al., 2018) and ReLVis (Saldanha et al., 2019), allow users to gain insight about the behavior pattern between training iterations in discrete and continuous action space, respectively. As the users explore the machine learning algorithms better, they can compare different methods and adjust the model faster.

In the meanwhile, increasing the understandability of machine learning makes those algorithms more trustworthy and actionable and extends the application to more areas. An example is visualizing CNNs for autonomous driving where visualization serves as a debugging tool for real-time CNN-based systems. This is done by visualizing the regions of the input image which have the highest influence on the output (Bojarski et al., 2016). A recent paper (Endert et al., 2017) gives a summary of literature on how interactive VA is involved with the machine learning domains of dimension reduction, clustering, classification, and regression. It also shows some application domains in the field of integrating machine learning with VA, text analytics, and biological data analytics.

Parameter Initialization

In essence, all the machining learning problems are optimization problems where we minimize the error/loss or maximize the benefit/probability from an initial start point. A bad initialization may lead to a slow converging path or even a sub-optimal result. In the following sections, we will introduce some works which apply human knowledge to initialization. We introduce how an agent learns from expert behaviors in Pre-training in Reinforcement Learning and how to use trained models by Transfer Learning. Although transfer learning is not limited to initialization, we categorize it here considering that fine-tuning pre-trained models is a widely used technique of transfer learning.

Pre-training in Reinforcement Learning

Many reinforcement problems must be solved in a large state/action space, especially for the continuous state action problems (Lazaric et al., 2008). Learning in a high-dimensional space requires huge amounts of data and learning time. Also, in many optimization methods, such as gradient-descent method and Newton's method, the initialization plays a critical role and may determine whether we are able to find the optimal value function/policy. Thus, equipping a pre-trained function as initialization in reinforcement learning becomes popular these days. Supervised learning is often used at the beginning stage of reinforcement learning, and the domain knowledge from humans is embedded in this way of initialization.

Pre-training can be applied to policy learning, value function learning, environment model learning, or a combination of them. In policy learning, humans can act as a trainer to teach agents the target parameterized policy through demonstration. A study provides a comprehensive survey of many different approaches to learning from demonstrations (Argall et al., 2009), and these approaches allow the agent to have a good initial policy before fine-tuning with interactive feedback. In the training of AlphaGO with DNNs and tree search (Silver et al., 2016), a supervised learning policy network is pre-trained directly from human expert moves. Then, a reinforce learning policy is trained to improve it by optimizing the final outcomes. In robot navigation, reinforce learning is capable of learning the fuzzy rules automatically but suffers from a heavy learning phase and insufficiently learned rules. Including supervised learning results as initialization in value function learning helps solve the issue and becomes one of the main approaches for this problem (Fathinezhad et al., 2016; Navarro-Guerrero et al., 2012; Ye et al., 2003). Learning a model of state dynamics can result in a pre-trained hidden layer structure that reduces the training time in reinforce learning problems (Anderson et al., 2015), and learning the deep Q networks from human demonstrators also helps to give a relatively good initial model and predict the dynamics (Gabriel et al., 2019). There are many other applications of smart initialization on policy gradient methods (Yun et al., 2017) and Q-learning methods (Burkov and Chaib-Draa, 2007; Song et al., 2012), which speed up the learning and level up the performance (Finn et al., 2016).

Transfer Learning

Transfer learning is where knowledge is learned from a "source task" and applied to a "target task" (Pan and Yang, 2009). This method is inherently dependent on human knowledge to determine suitable source tasks to transfer. The transferred knowledge can be data, neural networks, weights, etc. The ideal situation to use transfer learning is when the source task has more data available than the target task. However, transfer learning can yield benefits even when the source task does not have as much data. As shown in Figure 5, traditionally in machine learning, the data is specific to the task being trained. By transferring knowledge from the already trained source task, less data specific to the target task are needed and the training time is reduced. The context provided from the source task increases the initial performance, rate of improvement, and final performance (Brownlee, 2017) while reducing the training time and the data needed. This efficiency has led to significant commercial use of transfer learning (Ruder, 2017b).

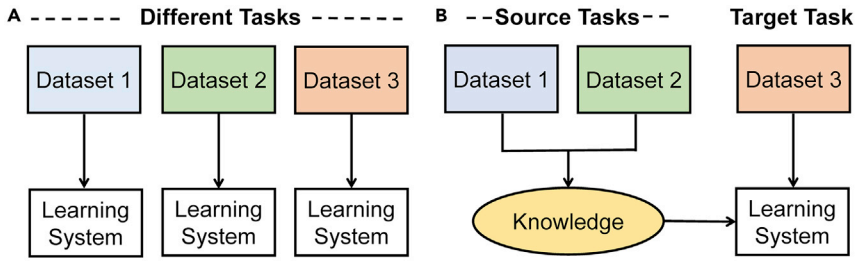

Figure 5. Illustration of Traditional Machine Learning and Transfer Learning

(A) Tasks in traditional machine learning do not share knowledge.

(B) Tasks in transfer learning share knowledge. Target task can reuse the knowledge of source tasks.

Drawn based on the concept in (Pan and Yang, 2009).

The role of human knowledge in transfer learning can be understood through the NLP problem of training a DNN for children's automatic speech recognition (ASR) (Shivakumar and Georgiou, 2020). There is a plethora of general speech data available, yet there is a lack of data specific to child speech recognition. It would be costly and time consuming to acquire data to train a child ASR algorithm from scratch. Instead, the data, neural networks, and weights from general ASR training can be transferred. Then, the weights can be fine-tuned for child ASR by using the limited amount of data specific to child ASR. The machine cannot know which source tasks would be beneficial to train the target task, so human knowledge is required to determine compatible and effective source tasks. Transfer learning is also used for image recognition where a human finds a source task to quickly train the target task before fine-tuning it with new visual data. This has been used in a wide variety of applications such as plant identification (Ghazi et al., 2017), structural damage detection (Gao and Mosalam, 2018; Gopalakrishnan et al., 2017), human behavior recognition (Kaya et al., 2017; Sargano et al., 2017), and even medical research (Burlina et al., 2017; Christodoulidis et al., 2016; Karri et al., 2017).

New methods of incorporating human knowledge with transfer learning are being researched to achieve higher efficiency and broader application. One of them is using simulation data. As we have mentioned in Section Simulation, machine learning can be trained with simulations before being fine-tuned with real-world data.

Another method that benefits from human knowledge is "heterogeneous transfer learning" (Day and Khoshgoftaar, 2017). Most commercial transfer learning methods are homogeneous, which means that the source and target tasks have the same feature space. For instance, in computer vision, the simulation looks different from the real world but the feature space is the same since they are both inputting pixels. In heterogeneous transfer learning, the feature spaces are different. For instance, the inputs are texts of two different languages, or one input is texts while the other is images. Heterogeneous transfer, by correlating different feature spaces, allows the source task to come from a wider variety of data. Extensive human knowledge on the subject is required to correlate the feature spaces effectively. The complexity of this correlation makes it difficult to scale heterogeneous transfer learning for broad use because each application of heterogeneous transfer learning requires a different subject of human knowledge. Other heterogeneous transfer learning techniques are being attempted to solve this problem, such as deep semantic mapping (Zhao et al., 2019) and hybrid heterogeneous transfer learning (Zhou et al., 2014, 2019).

Lastly, the importance of human knowledge for transfer learning is apparent by the poor performance that can occur without it. Sometimes the performance may actually be worse than if the target task was trained alone. This is known as negative transfer and occurs when the source tasks are poorly suited for the target tasks (Wang et al., 2019). It appears in human learning as well, e.g., learning to throw a baseball may be harder after learning to throw a football due to muscle memory making it difficult to adapt to a new throwing motion. Currently, preventing negative transfer requires effective human intuition or experience. Research is conducted to develop methods that will quantitatively eliminate negative performance, including using a discriminator gate to assign different weights to each source task (Wang et al., 2019) and using an iterative method that detects the source of the negative transfer to reduce class noise (Gui et al., 2018).

CONCLUSIONS

In this paper, we give a comprehensive review on integrating human knowledge into machine learning. Knowledge is categorized into general knowledge and domain knowledge, and its representations are introduced together with the works that leverage them. We focus on some new and popular topics and group the methods by their major contribution to the machine learning pipeline. In conclusion, based on existing methods, we propose the following suggestions on improving the machine learning performance with knowledge:

1. Devise the inputs and outputs of models to make better use of resources. Aggregate tasks to learn together by Multitask Learning if they share data or information; an auxiliary task can be attempted even if only a single task is important. Use Features that could best represent the essence of tasks; the features can be manually engineered, selected by statistic metrics, or automatically learned by machine learning models.
2. Examine model assumptions to capture major factors. Set Variable Relation such as independence based on prior knowledge. The Distribution of unknown variables in models can be obtained by empirical data, expert intuition or Gaussian. Try to match the distribution of training data with test scenarios.
3. If using neural networks, tailor the architecture to be suitable for the tasks. If possible, incorporate some known properties, such as Symmetry of Convolutional Neural Networks. Logic, equations, and temporal nature can be, respectively, reflected in the structure of networks by, for instance, combining with symbolic AI, designing special layers/architectures, and using RNNs.
4. Augment data to incorporate invariant properties or knowledge-based models. Augmentation can be done by transforming, manipulating, or synthesizing the original data. Image and Audio data have been discussed in details, and their augmentation share similar principles. Simulations built upon knowledge-based models can be used to generate data.
5. Design algorithms to include humans in the loop. The interaction between machine and environment can be modeled and optimized in Reinforcement Learning . Humans can be asked to label data or provide distribution (Active Learning). Interactive Visual Analytics can be used to help humans understand machine learning results and then adjust models during or after training.
6. Find better initialization to reflect known results. This could be achieved by learning from expert behaviors before allowing machine to automatically learn from the environment (Pre-training in Reinforcement Learning). Transfer Learning can be used to distill knowledge from relevant tasks.

For future works, we highlight the following directions:

1. Models are dedicated and specific to tasks rather than universal. We witnessed the emergence of CNNs for images and RNNs for natural language. They are intuitively and empirically better than fully connected networks. General knowledge inspires us to leverage math and brains to propose more efficient mechanisms, such as attention (Vaswani et al., 2017). Domain knowledge captures the nature of the tasks, and more customized components can be incorporated.
2. More nodes are added to end-to-end learning for human interaction, feedback, and intervention. Despite convenient data preparation, the black box approach of end-to-end learning makes it difficult to explain and control. We can regulate the intermediate results or network layers (Zhang et al., 2018) to produce models more understandable and controllable by humans.
3. Existing results are reused for new targets. Humans can use the skills and insights across multiple tasks and even disciplines; some abilities are innate. Similarly, machines do not need to be trained from scratch. Given a new task, we can transfer or distill the knowledge from previous tasks or models.
4. Higher level features such as conceptual understanding and math theorems are incorporated. Currently, the knowledge integrated in machine learning is relatively concrete and mostly at the instance level, e.g. expressing each theorem as a constraint. Despite the efforts and achievements to make knowledge generic and broad, we have not seen a successful model to grasp abstract concepts or systematic theories. We believe integrating higher level features is an essential path toward strong artificial intelligence and would change the paradigm to integrate knowledge.

Designing and implementing machine learning algorithms is an iterative process. This requires humans to analyze the models and knowledge integration to take advantage of human understanding of the real world. This review may help current and prospective users of machine learning to understand these fields and inspire them to build more efficient models.

ACKNOWLEDGMENTS

We gratefully acknowledge the support by the Ford Motor Company.

AUTHOR CONTRIBUTIONS

X.J. drafted Design of Neuron Connections, Reinforcement Learning, Interactive Visual Analytics, and Pre-training in Reinforcement Learning. C.R. drafted Multitask Learning, Simulation, and Transfer Learning. J.Z. drafted Qualitative Domain Knowledge and Quantitative Domain Knowledge. C.D. drafted other sections and edited the manuscript. W.L. supervised this review and revised the manuscript.

REFERENCES

Abdelaziz, I., Fokoue, A., Hassanzadeh, O., Zhang, P., and Sadoghi, M. (2017). Large-scale structural and textual similarity-based mining of knowledge graph to predict drug–drug interactions. J. Web Semant. *44*, 104–117.

Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B., and Rousseau, D. (2015). The Higgs boson machine learning challenge. In NIPS 2014 Workshop on High-Energy Physics and Machine Learning, pp. 19–55.

Afifi, M., and Brown, M.S. (2019). What else can fool deep learning? Addressing color constancy errors on deep neural network performance. In Proceedings of the IEEE International Conference on Computer Vision (IEEE), pp. 243–252.

Aha, D.W., and Bankert, R.L. (1996). A comparative evaluation of sequential feature selection algorithms. In Learning from Data (Springer), pp. 199–206.

Amos, B., and Kolter, J.Z. (2017). Optnet: differentiable optimization as a layer in neural networks. arXiv. arXiv:1703.00443.

Anderson, C.W., Lee, M., and Elliott, D.L. (2015). Faster reinforcement learning after pretraining deep networks to predict state dynamics. In 2015 International Joint Conference on Neural Networks (IEEE), pp. 1–7.

Argall, B.D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. Rob. Auton. Syst. *57*, 469–483.

Attenberg, J., Melville, P., and Provost, F. (2010). A unified approach to active dual supervision for labeling features and examples. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (Springer), pp. 40–55.

Bachman, P., Sordoni, A., and Trischler, A. (2018). Learning algorithms for active learning. In Proceedings of Machine Learning Research. arXiv:1708.00088.

Bahnsen, A.C., Aouada, D., Stojanovic, A., and Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. Expert Syst. Appl. *51*, 134–142.

Bair, E., and Tibshirani, R. (2004). Semi-supervised methods to predict patient survival from gene expression data. Plos Biol. *2*, e108.

Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In Proceedings of ICML Workshop on Unsupervised and Transfer Learning, pp. 37–49.

Baram, Y., Yaniv, R.E., and Luz, K. (2004). Online choice of active learning algorithms. J. Mach. Learn. Res. *5*, 255–291.

Barshan, E., Ghodsi, A., Azimifar, Z., and Jahromi, M.Z. (2011). Supervised principal component analysis: visualization, classification and regression on subspaces and submanifolds. Pattern Recognit *44*, 1357–1371.

Bartolozzi, C., and Indiveri, G. (2007). Synaptic dynamics in analog VLSI. Neural Comput. *19*, 2581–2603.

Bau, D., Zhu, J.-Y., Strobelt, H., Zhou, B., Tenenbaum, J.B., Freeman, W.T., and Torralba, A. (2019). Visualizing and understanding generative adversarial networks. arXiv. arXiv:1901.09887.

Belkin, M., and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. *15*, 1373–1396.

Bengio, Y., Courville, A.C., and Vincent, P. (2013). Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. *35*, 1798–1828.

Benjamin, B.V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A.R., Bussat, J.-M., Alvarez-Icaza, R., Arthur, J.V., Merolla, P.A., and Boahen, K. (2014). Neurogrid: a mixed-analog-digital multichip system for large-scale neural simulations. Proc. IEEE. *102*, 699–716.

Bojarski, M., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., Muller, U., and Zieba, K. (2016). Visualbackprop: efficient visualization of CNNS. arXiv. arXiv:1611.05418.

Boluki, S., Esfahani, M.S., Qian, X., and Dougherty, E.R. (2017). Incorporating biological prior knowledge for Bayesian learning via maximal knowledge-driven information priors. BMC Bioinformatics *18*, 552.

Brause, R., Langsdorf, T., and Hepp, M. (1999). Neural data mining for credit card fraud detection. In Proceedings of the 11th International Conference on Tools with Artificial Intelligence (IEEE)), pp. 103–106.

Brownlee, J. (2014). Discover Feature Engineering, How to Engineer Features and How to Get Good at it. https://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/.

Brownlee, J. (2017). A Gentle Introduction to Transfer Learning for Deep Learning. https://machinelearningmastery.com/transfer-learning-for-deep-learning/.

Burbidge, R., Rowland, J.J., and King, R.D. (2007). Active learning for regression based on query by committee. In International Conference on Intelligent Data Engineering and Automated Learning (Springer), pp. 209–218.

Burkov, A., and Chaib-Draa, B. (2007). Reducing the complexity of multiagent reinforcement learning. In Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1–3.

Burlina, P., Pacheco, K.D., Joshi, N., Freund, D.E., and Bressler, N.M. (2017). Comparing humans and deep learning performance for grading AMD: a study in using universal deep features and transfer learning for automated AMD analysis. Comput. Biol. Med. *82*, 80–86.

Cao, Y., Wang, X., He, X., Hu, Z., and Chua, T.-S. (2019). Unifying knowledge graph learning and recommendation: towards a better understanding of user preferences. In The World Wide Web Conference, pp. 151–161.

Chandrashekar, G., and Sahin, F. (2014). A survey on feature selection methods. Comput. Electr. Eng. *40*, 16–28.

Chawla, N.V., Bowyer, K.W., Hall, L.O., and Kegelmeyer, W.P. (2002). SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. *16*, 321–357.

Chen, C., Zhang, L., Bu, J., Wang, C., and Chen, W. (2010). Constrained Laplacian eigenmap for dimensionality reduction. Neurocomputing *73*, 951–958.

Chen, N.C., Drouhard, M., Kocielnik, R., Suh, J., and Aragon, C.R. (2018). Using machine learning to support qualitative coding in social science: shifting the focus to ambiguity. ACM TiiS *8*, 1–20.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. arXiv. arXiv:2002.05709.

Choo, J., and Liu, S. (2018). Visual analytics for explainable deep learning. IEEE Comput. Graph Appl. *38*, 84–92.

Choo, J., Lee, H., Kihm, J., and Park, H. (2010). iVisClassifier: an interactive visual analytics system for classification based on supervised dimension reduction. In 2010 IEEE Symposium on Visual Analytics Science and Technology (IEEE), pp. 27–34.

Christodoulidis, S., Anthimopoulos, M., Ebner, L., Christe, A., and Mougiakakou, S. (2016). Multisource transfer learning with convolutional neural networks for lung pattern analysis. IEEE J. Biomed. Health Inform. *21*, 76–84.

Cohen, T., and Welling, M. (2016). Group equivariant convolutional networks. In International Conference on Machine Learning, pp. 2990–2999.

Crowston, K., Allen, E.E., and Heckman, R. (2012). Using natural language processing technology for qualitative data analysis. Int. J. Soc. Res. Methodol. *15*, 523–543.

Daniušis, P., Vaitkus, P., and Petkevičius, L. (2016). Hilbert–Schmidt component analysis. Proc. Lith. Math. Soc. Ser. A. *57*, 7–11.

Dasgupta, S., and Hsu, D. (2008). Hierarchical sampling for active learning. In Proceedings of the 25th International Conference on Machine Learning, pp. 208–215.

Dash, M., and Liu, H. (2003). Consistency-based search in feature selection. Artif. Intell. *151*, 155–176.

Day, O., and Khoshgoftaar, T.M. (2017). A survey on heterogeneous transfer learning. J. Big Data *4*, 29.

DeBrusk, C. (2018). The Risk of Machine-Learning Bias (And How to Prevent it) (MIT Sloan Management Review).

Davies, M., Srinivasa, N., Lin, T.H., Chinya, G., Cao, Y., Chody, S.H., Dimou, G., Joshi, P., Imam, N., Jain, S., et al. (2018). Loihi: a neuromorphic manycore processor with on-chip learning. IEEE Micro *38*, 82–99.

Deng, C., Wang, Y., Qin, C., and Lu, W. (2020). Self-directed online machine learning for topology optimization. arXiv. arXiv:2002.01927.

Deng, J., Ding, N., Jia, Y., Frome, A., Murphy, K., Bengio, S., Li, Y., Neven, H., and Adam, H. (2014). Large-scale object classification using label relation graphs. In European Conference on Computer Vision (Springer), pp. 48–64.

Deshmukh, A., Morghade, J., Khera, A., and Bajaj, P. (2005). Binary neural networks–a CMOS design approach. In International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (Springer), pp. 1291–1296.

DeVries, T., and Taylor, G.W. (2017). Dataset augmentation in feature space. arXiv. arXiv:1702.05538.

Dhanjal, C., Gunn, S.R., and Shawe-Taylor, J. (2008). Efficient sparse kernel feature extraction based on partial least squares. IEEE Trans. Pattern Anal. Mach. Intell. *31*, 1347–1361.

Díaz, I., Cuadrado, A.A., and Verleysen, M. (2016). A statespace model on interactive dimensionality reduction. In 24th European Symposium on Artificial Neural Networks, pp. 647–652.

Drachman, D.A. (2005). Do we have brain to spare? Neurology *64*, 2004–2005.

Druck, G., Settles, B., and McCallum, A. (2009). Active learning by labeling features. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pp. 81–90.

Du, B., Wang, Z., Zhang, L., Zhang, L., Liu, W., Shen, J., and Tao, D. (2015). Exploring representativeness and informativeness for active learning. IEEE Trans. Cybern. *47*, 14–26.

Ebden, M. (2015). Gaussian processes: a quick introduction. arXiv. arXiv:1505.02965.

Ebert, S., Fritz, M., and Schiele, B. (2012). Ralf: a reinforced active learning formulation for object class recognition. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3626–3633.

Ehrlich, M., Shields, T.J., Almaev, T., and Amer, M.R. (2016). Facial attributes classification using multi-task representation learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 47–55.

Endert, A., Fiaux, P., and North, C. (2012). Semantic interaction for visual text analytics. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 473–482.

Endert, A., Ribarsky, W., Turkay, C., Wong, B.W., Nabney, I., Blanco, I.D., and Rossi, F. (2017). The state of the art in integrating machine learning into visual analytics. Comput. Graphics Forum *36*, 458–486.

Ermon, S., Bras, R.L., Suram, S.K., Gregoire, J.M., Gomes, C., Selman, B., and Dover, R.B.V. (2014). Pattern decomposition with complex combinatorial constraints: application to materials discovery. arXiv. arXiv:1411.7441.

Fails, J.A., and Olsen, D.R. (2003). Interactive machine learning. In Proceedings of the 8th International Conference on Intelligent User Interfaces, pp. 39–45.

Fadaee, M., Bisazza, A., and Monz, C. (2017). Data augmentation for low-resource neural machine translation. arXiv. arXiv:1705.00440.

Farahmand, A., Nabi, S., and Nikovski, D.N. (2017). Deep reinforcement learning for partial differential equation control. In 2017 American Control Conference (IEEE), pp. 3120–3127.

Fathinezhad, F., Derhami, V., and Rezaeian, M. (2016). Supervised fuzzy reinforcement learning for robot navigation. Appl. Soft Comput. *40*, 33–41.

Fellbaum, C. (2012). WordNet. In The Encyclopedia of Applied Linguistics, C. Chapelle, ed. (Blackwell Publishing Ltd.), pp. 1–8.

Finn, C., Yu, T., Fu, J., Abbeel, P., and Levine, S. (2016). Generalizing skills with semi-supervised reinforcement learning. arXiv. arXiv:1612.00429.

Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems. Ann. Eugen. *7*, 179–188.

Flasiński, M. (2016). Symbolic artificial intelligence. Introduction to Artificial Intelligence (Springer), pp. 15–22.

Flores, M.J., Nicholson, A.E., Brunskill, A., Korb, K.B., and Mascaro, S. (2011). Incorporating expert knowledge when learning Bayesian network structure: a medical case study. Artif. Intell. Med. *53*, 181–204.

Fogg, A. (2017). A History of Machine Learning and Deep Learning. https://www.import.io/post/history-of-deep-learning/.

Frank, E., and Bouckaert, R.R. (2006). Naive Bayes for text classification with unbalanced classes. In European Conference on Principles of Data Mining and Knowledge Discovery (Springer), pp. 503–510.

Frohlich, H., Chapelle, O., and Scholkopf, B. (2003). Feature selection for support vector machines by means of genetic algorithm. In Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence, pp. 142–148.

Fu, L.M. (1993). Knowledge-based connectionism for revising domain theories. IEEE Trans. Syst. Man. Cybern. Syst. *23*, 173–182.

Gabriel, V., Du, Y., and Taylor, M.E. (2019). Pre-training with non-expert human demonstration for deep reinforcement learning. Knowl. Eng. Rev. *34*, e10.

Gal, Y., and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In International Conference on Machine Learning, pp. 1050–1059.

Gal, Y., Islam, R., and Ghahramani, Z. (2017). Deep Bayesian active learning with image data. arXiv. arXiv:1703.02910.

Gan, J., Cai, Q., Galer, P., Ma, D., Chen, X., Huang, J., Bao, S., and Luo, R. (2019). Mapping the knowledge structure and trends of epilepsy genetics over the past decade: a co-word analysis based on medical subject headings terms. Medicine *98*, e16782.

Gao, T., and Lu, W. (2020). Physical model and machine learning enabled electrolyte channel design for fast charging. J. Electrochem. Soc. *167*, 110519.

Gao, Y., and Mosalam, K.M. (2018). Deep transfer learning for image-based structural damage recognition. Comput. Aided Civil Infrastruct. Eng. *33*, 748–768.

Garnelo, M., and Shanahan, M. (2019). Reconciling deep learning with symbolic artificial intelligence: representing objects and relations. Curr. Opin. Behav. Sci. *29*, 17–23.

Gatys, L.A., Ecker, A.S., and Bethge, M. (2015). A neural algorithm of artistic style. arXiv. arXiv:1508.06576.

Gens, R., and Domingos, P.M. (2014). Deep symmetry networks. In Advances in Neural Information Processing Systems, pp. 2537–2545.

Ghazi, M.M., Yanikoglu, B., and Aptoula, E. (2017). Plant identification using deep neural networks via optimization of transfer learning parameters. Neurocomputing *235*, 228–235.

Ghojogh, B., Samad, M.N., Mashhadi, S.A., Kapoor, T., Ali, W., Karray, F., and Crowley, M. (2019). Feature selection and feature extraction in pattern analysis: a literature review. arXiv. arXiv:1905.02845.

Ghosh, R., and Gupta, A.K. (2019). Scale steerable filters for locally scale-invariant convolutional neural networks. arXiv. arXiv:1906.03861.

Giffard-Roisin, S., Delingette, H., Jackson, T., Webb, J., Fovargue, L., Lee, J., Rinaldi, C.A., Razavi, R., Ayache, N., and Sermesant, M. (2018). Transfer learning from simulations on a reference anatomy for ECGI in personalized cardiac resynchronization therapy. IEEE. Trans. Biomed. Eng. *66*, 343–353.

Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448.

Gong, T., Lee, T., Stephenson, C., Renduchintala, V., Padhy, S., Ndirango, A., Keskin, G., and Elibol, O.H. (2019). A comparison of loss weighting strategies for multi task learning in deep neural networks. IEEE Access *7*, 141627–141632.

Goodfellow, I.J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv. arXiv:1412.6572.

Gopalakrishnan, K., Khaitan, S.K., Choudhary, A., and Agrawal, A. (2017). Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. Constr. Build Mater. *157*, 322–330.

Gori, M., Monfardini, G., and Scarselli, F. (2005). A new model for learning in graph domains. In Proceedings of 2005 IEEE International Joint Conference on Neural Networks, pp. 729–734.

Graves, A., Mohamed, A., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649.

Griffith, S., Subramanian, K., Scholz, J., Isbell, C.L., and Thomaz, A.L. (2013). Policy shaping: integrating human feedback with reinforcement learning. In Advances in Neural Information Processing Systems, C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, eds. (Curran Associates), pp. 2625–2633.

Gui, L., Xu, R., Lu, Q., Du, J., and Zhou, Y. (2018). Negative transfer detection in transductive transfer learning. Int. J. Mach. Learn. Cybern. *9*, 185–197.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9729–9738.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Hennecke, M., Frings, W., Homberg, W., Zitz, A., Knobloch, M., and Böttiger, H. (2012). Measuring power consumption on IBM Blue gene/P. Comput. Sci. Res. Dev. *27*, 329–336.

Herculano-Houzel, S. (2009). The human brain in numbers: a linearly scaled-up primate brain. Front. Hum. Neurosci. *3*, 31.

Hinton, G.E., Krizhevsky, A., and Wang, S.D. (2011). Transforming auto-encoders. In International Conference on Artificial Neural Networks (Springer), pp. 44–51.

Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv. arXiv:1207.0580.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. Neural Comput. *9*, 1735–1780.

Hodgkin, A.L., and Huxley, A.F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. *117*, 500.

Hoehndorf, R., and Queralt-Rosinach, N. (2017). Data science and symbolic AI: synergies, challenges and opportunities. Data Sci. *1*, 27–38.

Hohman, F., Kahng, M., Pienta, R., and Chau, D.H. (2018). Visual analytics in deep learning: an interrogative survey for the next frontiers. IEEE Trans. Vis. Comput. Graph. *25*, 2674–2693.

Hoi, S.C., Jin, R., Zhu, J., and Lyu, M.R. (2009). Semisupervised SVM batch mode active learning with applications to image retrieval. ACM Trans. Inf. Syst. *27*, 1–29.

Holmes, T.W., Ma, K., and Pourmorteza, A. (2019). Combination of CT motion simulation and deep convolutional neural networks with transfer learning to recover Agatston scores. In 15th International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine (International Society for Optics and Photonics), p. 110721Z.

Holzinger, A. (2016). Interactive machine learning for health informatics: when do we need the human-in-the-loop? Brain Inform. *3*, 119–131.

Holzinger, A., Kieseberg, P., Weippl, E., and Tjoa, A.M. (2018). Current advances, trends and challenges of machine learning and knowledge extraction: from machine learning to explainable AI. In International Cross-Domain Conference for Machine Learning and Knowledge Extraction (Springer), pp. 1–8.

Holzinger, A., Plass, M., Kickmeier-Rust, M., Holzinger, K., Crişan, G.C., Pintea, C.-M., and Palade, V. (2019). Interactive machine learning: experimental evidence for the human in the algorithmic loop. Appl. Intell. *49*, 2401–2414.

Holzinger, A., Stocker, C., Ofner, B., Prohaska, G., Brabenetz, A., and Hofmann-Wellenhof, R. (2013). Combining HCI, natural language processing, and knowledge discovery-potential of IBM content analytics as an assistive technology in the biomedical field. In International Workshop on Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data (Springer), pp. 13–24.

Hu, Y., Huber, A., Anumula, J., and Liu, S.-C. (2018). Overcoming the vanishing gradient problem in plain recurrent networks. arXiv. arXiv:1801.06105.

Hu, Y., Nakhaei, A., Tomizuka, M., and Fujimura, K. (2019). Interaction-aware decision making with adaptive strategies under merging scenarios. arXiv. arXiv:1904.06025.

Hu, Z., Ma, X., Liu, Z., Hovy, E., and Xing, E. (2016). Harnessing deep neural networks with logic rules. arXiv. arXiv:1603.06318.

Huang, S.-J., Jin, R., and Zhou, Z.-H. (2010). Active learning by querying informative and representative examples. In Advances in Neural Information Processing Systems, J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, eds. (Curran Associates), pp. 892–900.

Hwang, Y., Cho, H., Yang, H., Oh, I., and Lee, S.-W. (2020). Mel-spectrogram augmentation for sequence to sequence voice conversion. arXiv. arXiv:2001.01401.

Hyvärinen, A. (2013). Independent component analysis: recent advances. Philos. T. R. Soc. A. *371*, 20110534.

Ikebata, H., Hongo, K., Isomura, T., Maezono, R., and Yoshida, R. (2017). Bayesian molecular design with a chemical language model. J. Comput. Aided Mol. Des. *31*, 379–391.

Inoue, H. (2018). Data augmentation by pairing samples for images classification. arXiv. arXiv:1801.02929.

Jackson, P.T., Abarghouei, A.A., Bonner, S., Breckon, T.P., and Obara, B. (2019). Style augmentation: data augmentation via style randomization. In CVPR Workshops, pp. 83–92.

Jeong, D.H., Ziemkiewicz, C., Fisher, B., Ribarsky, W., and Chang, R. (2009). iPCA: An Interactive System for PCA-based Visual Analytics. Comput. Graphics Forum *28*, 767–774.

Ji, X.A., Molnar, T.G., Avedisov, S.S., and Orosz, G. (2020). Feed-forward neural network with trainable delay. In Proceedings of the 2nd Conference on Learning for Dynamics and Control, A. Bayen, A. Jadbabaie, G.J. Pappas, P. Parrilo, B. Recht, C. Tomlin, and M. Zeilinger, eds. (PLMR), pp. 127–136.

Jurio, A., Pagola, M., Galar, M., Lopez-Molina, C., and Paternain, D. (2010). A comparison study of different color spaces in clustering based image segmentation. In International Conference on Information Processing and Management of

Uncertainty in Knowledge-Based Systems (Springer), pp. 532–541.

Kanazawa, A., Sharma, A., and Jacobs, D. (2014). Locally scale-invariant convolutional neural networks. arXiv. arXiv:1412.5104.

Kang, G., Dong, X., Zheng, L., and Yang, Y. (2017). Patchshuffle regularization. arXiv. arXiv:1707.07103.

Karpathy, A., Johnson, J., and Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. arXiv. arXiv:1506.02078.

Karri, S.P.K., Chakraborty, D., and Chatterjee, J. (2017). Transfer learning based classification of optical coherence tomography images with diabetic macular edema and dry age-related macular degeneration. Biomed. Opt. Express *8*, 579–592.

Kaya, H., Gürpınar, F., and Salah, A.A. (2017). Video-based emotion recognition in the wild using deep transfer learning and score fusion. Image Vis. Comput *65*, 66–75.

Kelley, H.J. (1960). Gradient theory of optimal flight paths. ARS J. *30*, 947–954.

Knox, W.B., and Stone, P. (2008). Tamer: training an agent manually via evaluative reinforcement. In 2008 7th IEEE International Conference on Development and Learning, pp. 292–297.

Knox, W.B., and Stone, P. (2010). Combining manual feedback with subsequent MDP reward signals for reinforcement learning. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, pp. 5–12.

Knox, W.B., and Stone, P. (2012). Reinforcement learning from simultaneous human and MDP reward. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, pp. 475–482.

Kok, J.R., and Vlassis, N. (2004). Sparse tabular multiagent Q-learning. In Annual Machine Learning Conference of Belgium and the Netherlands, pp. 65–71.

Konyushkova, K., Sznitman, R., and Fua, P. (2017). Learning active learning from data. In Advances in Neural Information Processing Systems, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds. (Curran Associates), pp. 4225–4235.

Kromp, F., Ambros, I., Weiss, T., Bogen, D., Dodig, H., Berneder, M., Gerber, T., Taschner-Mandl, S., Ambros, P., and Hanbury, A. (2016). Machine learning framework incorporating expert knowledge in tissue image annotation. In 2016 International Conference on Pattern Recognition (IEEE), pp. 343–348.

Kursa, M.B., and Rudnicki, W.R. (2010). Feature selection with the Boruta package. J. Stat. Softw. *36*, 1–13.

Lazaric, A., Restelli, M., and Bonarini, A. (2008). Reinforcement learning in continuous action spaces through sequential Monte Carlo methods. In Advances in Neural Information Processing Systems, J.C. Platt, D. Koller, Y. Singer, and S.T. Roweis, eds. (Curran Associates), pp. 833–840.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature *521*, 436–444.

Lee, K., and Kim, D. (2019). In-silico molecular binding prediction for human drug targets using deep neural multi-task learning. Genes *10*, 906.

Li, X., Zhao, L., Wei, L., Yang, M.-H., Wu, F., Zhuang, Y., Ling, H., and Wang, J. (2016). Deepsaliency: multi-task deep neural network model for salient object detection. IEEE Trans. Image Process. *25*, 3919–3930.

Li, Y., Min, M.R., Shen, D., Carlson, D.E., and Carin, L. (2018). Video generation from text. In AAAI Conference on Artificial Intelligence, pp. 7065–7072.

Liem, C.C., Langer, M., Demetriou, A., Hiemstra, A.M., Wicaksana, A.S., Born, M.P., and König, C.J. (2018). Psychology meets machine learning: interdisciplinary perspectives on algorithmic job candidate screening. In Explainable and Interpretable Models in Computer Vision and Machine Learning, H. Escalante, S. Escalera, I. Guyon, X. Baró, Y. Güçlütürk, U. Güçlü, and M.V. Gerven, eds. (Springer), pp. 197–253.

Lin, Z., Harrison, B., Keech, A., and Riedl, M.O. (2017). Explore, exploit or listen: combining human feedback and policy model to speed up deep reinforcement learning in 3d worlds. arXiv. arXiv:1709.03969.

Lin, Z., Shi, Y., and Xue, Z. (2018). IDSGAN: generative adversarial networks for attack generation against intrusion detection. arXiv. arXiv:1809.02077.

Ling, J., Jones, R., and Templeton, J. (2016). Machine learning strategies for systems with invariance properties. J. Comput. Phys. *318*, 22–35.

Liu, Z., Wang, J., Gong, S., Lu, H., and Tao, D. (2019). Deep reinforcement active learning for human-in-the-loop person re-identification. In Proceedings of the IEEE International Conference on Computer Vision, pp. 6122–6131.

Loftin, R., Peng, B., MacGlashan, J., Littman, M.L., Taylor, M.E., Huang, J., and Roberts, D.L. (2016). Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning. Auton. Agent Multi Agent Syst. *30*, 30–59.

Long, M., Cao, Z., Wang, J., and Philip, S.Y. (2017). Learning multiple tasks with multilinear relationship networks. In Advances in Neural Information Processing Systems, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds. (Curran Associates), pp. 1594–1603.

Lowe, H.J., and Barnett, G.O. (1994). Understanding and using the medical subject headings (MeSH) vocabulary to perform literature searches. JAMA *271*, 1103–1108.

Ma, E. (2019). Data Augmentation in NLP. https://towardsdatascience.com/data-augmentation-in-nlp-2801a34dfc28.

Maaten, L.van der, and Hinton, G. (2008). Visualizing data using t-SNE. J. Mach. Learn. Res. *9*, 2579–2605.

MacGlashan, J., Ho, M.K., Loftin, R., Peng, B., Roberts, D., Taylor, M.E., and Littman, M.L. (2017). Interactive learning from policy-dependent human feedback. arXiv. arXiv:1701.06049.

Mafarja, M.M., Eleyan, D., Jaber, I., Hammouri, A., and Mirjalili, S. (2017). Binary dragonfly algorithm for feature selection. In 2017 International Conference on New Trends in Computing Sciences (IEEE), pp. 12–17.

Mann, G.S., and McCallum, A. (2010). Generalized expectation criteria for semi-supervised learning with weakly labeled data. J. Mach. Learn. Res. *11*, 955–984.

Mao, J., Gan, C., Kohli, P., Tenenbaum, J.B., and Wu, J. (2019). The neuro-symbolic concept learner: interpreting scenes, words, and sentences from natural supervision. arXiv. arXiv:1904.12584.

Martinez, M., Sitawarin, C., Finch, K., Meincke, L., Yablonski, A., and Kornhauser, A. (2017). Beyond grand theft auto V for training, testing and enhancing deep learning in self driving cars. arXiv. arXiv:1712.01397.

Masegosa, A.R., and Moral, S. (2013). An interactive approach for Bayesian network learning using domain/expert knowledge. Int. J. Approx. Reason. *54*, 1168–1181.

Mayr, A., Klambauer, G., Unterthiner, T., and Hochreiter, S. (2016). DeepTox: toxicity prediction using deep learning. Front. Environ. Sci. *3*, 80.

McCulloch, W.S., and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biol. *5*, 115–133.

Merolla, P.A., Arthur, J.V., Alvarez-Icaza, R., Cassidy, A.S., Sawada, J., Akopyan, F., Jackson, B.L., Imam, N., Guo, C., and Nakamura, Y. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. Science *345*, 668–673.

Meyerson, E., and Miikkulainen, R. (2018). Pseudo-task augmentation: from deep multitask learning to intratask sharing-and back. arXiv. arXiv:1803.04062.

Mignot, R., and Peeters, G. (2019). An analysis of the effect of data augmentation methods: experiments for a musical genre classification task. Trans. Int. Soc. Music Inf. Retr. *2*, 97–110.

Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Mullers, K.-R. (1999). Fisher discriminant analysis with kernels. In Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (IEEE), pp. 41–48.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. (2016). Cross-stitch networks for multi-task learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3994–4003.

Modha, D.S. (2017). Introducing a Brain-Inspired Computer. https://www.research.ibm.com/articles/brain-chip.shtml.

Mor, B., Garhwal, S., and Kumar, A. (2020). A systematic review of hidden markov models and

their applications. Arch. Comput. Methods Eng. https://doi.org/10.1007/s11831-020-09422-4.

Murphy, K.P. (2012). Machine Learning: A Probabilistic Perspective (MIT press).

Nakamura, K., Kuo, W.-J., Pegado, F., Cohen, L., Tzeng, O.J., and Dehaene, S. (2012a). Universal brain systems for recognizing word shapes and handwriting gestures during reading. Proc. Natl. Acad. Sci. *109*, 20762–20767.

Nakamura, R.Y., Pereira, L.A., Costa, K.A., Rodrigues, D., Papa, J.P., and Yang, X.-S. (2012b). BBA: a binary bat algorithm for feature selection. In 25th SIBGRAPI Conference on Graphics, Patterns and Images (IEEE), pp. 291–297.

Nanni, L., Maguolo, G., and Paci, M. (2020). Data augmentation approaches for improving animal audio classification. Ecol. Inform. *57*, 101084.

Navarro-Guerrero, N., Weber, C., Schroeter, P., and Wermter, S. (2012). Real-world reinforcement learning for autonomous humanoid robot docking. Rob. Auton. Syst. *60*, 1400–1407.

Nawrocki, R.A., Voyles, R.M., and Shaheen, S.E. (2016). A mini review of neuromorphic architectures and implementations. IEEE Trans. Electron. Devices *63*, 3819–3829.

Nguyen, H.T., and Smeulders, A. (2004). Active learning using pre-clustering. In Proceedings of the 21st International Conference on Machine Learning, p. 79.

Oord, A.van den, Kalchbrenner, N., Espeholt, L., Vinyals, O., and Graves, A. (2016b). Conditional image generation with pixelcnn decoders. In Advances in Neural Information Processing Systems, D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, and R. Garnett, eds. (Curran Associates), pp. 4790–4798.

Oord, A.van den, Kalchbrenner, N., and Kavukcuoglu, K. (2016a). Pixel recurrent neural networks. arXiv. arXiv:1601.06759.

Pan, S.J., and Yang, Q. (2009). A survey on transfer learning. IEEE Trans. Knowl. Data Eng. *22*, 1345–1359.

Pang, K., Dong, M., Wu, Y., and Hospedales, T. (2018). Meta-learning transferable active learning policies by deep reinforcement learning. arXiv. arXiv:1806.04798.

Parish, E.J., and Duraisamy, K. (2016). A paradigm for data-driven predictive modeling using field inversion and machine learning. J. Comput. Phys. *305*, 758–774.

Park, D.S. (2019). SpecAugment: A New Data Augmentation Method for Automatic Speech Recognition. https://ai.googleblog.com/2019/04/specaugment-new-data-augmentation.html.

Paulheim, H. (2017). Knowledge graph refinement: a survey of approaches and evaluation methods. Semantic web *8*, 489–508.

Peters, B.G. (2019). Institutional Theory in Political Science: The New Institutionalism (Edward Elgar Publishing).

Qu, M., Bengio, Y., and Tang, J. (2019). GMNN: graph Markov neural networks. arXiv. arXiv:1905.06214.

Raghavan, H., Madani, O., and Jones, R. (2006). Active learning with feedback on features and instances. J. Mach. Learn. Res. *7*, 1655–1686.

Raissi, M., Perdikaris, P., and Karniadakis, G.E. (2017). Machine learning of linear differential equations using Gaussian processes. J. Comput. Phys. *348*, 683–693.

Ramamurthy, R., Bauckhage, C., Sifa, R., Schücker, J., and Wrobel, S. (2019). Leveraging domain knowledge for reinforcement learning using MMC architectures. In International Conference on Artificial Neural Networks (Springer), pp. 595–607.

Ramires, A., and Serra, X. (2019). Data augmentation for instrument classification robust to audio effects. arXiv. arXiv:1907.08520.

Ramsundar, B., Kearnes, S., Riley, P., Webster, D., Konerding, D., and Pande, V. (2015). Massively multitask networks for drug discovery. arXiv. arXiv:1502.02072.

Ranjan, R., Patel, V.M., and Chellappa, R. (2017). Hyperface: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. IEEE Trans. Pattern Anal. Mach. Intell. *41*, 121–135.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. arXiv. arXiv:1605.05396.

Rennie, J.D., Shih, L., Teevan, J., and Karger, D.R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In Proceedings of the 20th International Conference on Machine Learning, pp. 616–623.

Ritter, S., Barrett, D.G., Santoro, A., and Botvinick, M.M. (2017). Cognitive psychology for deep neural networks: a shape bias case study. arXiv. arXiv:1706.08606.

Ritzer, G., and Stepnisky, J. (2017). Modern Sociological Theory (Sage publications).

Rong, X., and Adar, E. (2016). Visual tools for debugging neural language models. In Proceedings of ICML Workshop on Visualization for Deep Learning.

Rosenblatt, F. (1957). The Perceptron, a Perceiving and Recognizing Automaton (Cornell Aeronautical Laboratory).

Rosenfeld, A., Cohen, M., Taylor, M.E., and Kraus, S. (2018). Leveraging human knowledge in tabular reinforcement learning: a study of human subjects. Knowl. Eng. Rev. *33*, e14.

Ruder, S. (2017a). An overview of multi-task learning in deep neural networks. arXiv. arXiv:1706.05098.

Ruder, S. (2017b). Transfer Learning-Machine Learning's Next Frontier. https://ruder.io/transfer-learning/.

Ruder, S., Bingel, J., Augenstein, I., and Søgaard, A. (2019). Latent multi-task architecture learning. In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4822–4829.

Rueden, L.von, Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., Kirsch, B., Pfrommer, J., Pick, A., and Ramamurthy, R. (2019). Informed machine learning-A taxonomy and survey of integrating knowledge into learning systems. arXiv. arXiv:1903.12394.

Ruiz, N., Schulter, S., and Chandraker, M. (2018). Learning to simulate. arXiv. arXiv:1810.02513.

Sacha, D., Zhang, L., Sedlmair, M., Lee, J.A., Peltonen, J., Weiskopf, D., North, S.C., and Keim, D.A. (2016). Visual interaction with dimensionality reduction: a structured literature analysis. IEEE Trans. Vis. Comput. Graph *23*, 241–250.

Saito, P.T., Suzuki, C.T., Gomes, J.F., Rezende, P.J.de, and Falcão, A.X. (2015). Robust active learning for the diagnosis of parasites. Pattern Recognit *48*, 3572–3583.

Salakhutdinov, R., and Hinton, G. (2009). Deep Boltzmann machines. In Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, D.V. Dyk and M. Welling, eds. (PMLR), pp. 448–455.

Salamon, J., and Bello, J.P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal Process. Lett. *24*, 279–283.

Saldanha, E., Praggastis, B., Billow, T., and Arendt, D.L. (2019). ReLVis: visual analytics for situational awareness during reinforcement learning experimentation. In EuroVis, pp. 43–47.

Samaniego, E., Anitescu, C., Goswami, S., Nguyen-Thanh, V.M., Guo, H., Hamdia, K., Zhuang, X., and Rabczuk, T. (2020). An energy approach to the solution of partial differential equations in computational mechanics via machine learning: concepts, implementation and applications. Comput. Methods Appl. Mech. Eng. *362*, 112790.

Sargano, A.B., Wang, X., Angelov, P., and Habib, Z. (2017). Human action recognition using transfer learning with deep representations. In 2017 International Joint Conference on Neural Networks (IEEE), pp. 463–469.

Segler, M.H., Preuss, M., and Waller, M.P. (2018). Planning chemical syntheses with deep neural networks and symbolic AI. Nature *555*, 604–610.

Senior, A.W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A.W., and Bridgland, A. (2020). Improved protein structure prediction using potentials from deep learning. Nature *577*, 706–710.

Settles, B. (2011). Closing the loop: fast, interactive semi-supervised annotation with queries on features and instances. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1467–1478.

Settles, B. (2012). Active learning. Synth. Lect. Artif. Intell. Mach. Learn. *6*, 1–114.

Settles, B., and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pp. 1070–1079.

Shah, A., Wilson, A., and Ghahramani, Z. (2014). Student-t processes as alternatives to Gaussian processes. In Proceedings of the Seventeenth

International Conference on Artificial Intelligence and Statistics, S. Kaski and J. Corander, eds. (PLMR), pp. 877–885.

Shental, N., Bar-Hillel, A., Hertz, T., and Weinshall, D. (2004). Computing Gaussian mixture models with EM using equivalence constraints. In Advances in Neural Information Processing Systems, T. Thrun, L.K. Saul, and B. Schölkopf, eds. (MIT Press), pp. 465–472.

Shi, G., Shi, X., O'Connell, M., Yu, R., Azizzadenesheli, K., Anandkumar, A., Yue, Y., and Chung, S.J. (2019). Neural lander: stable drone landing control using learned dynamics. In 2019 International Conference on Robotics and Automation (IEEE), pp. 9784–9790.

Shivakumar, P.G., and Georgiou, P. (2020). Transfer learning from adult to children for speech recognition: evaluation, analysis and recommendations. Comput. Speech Lang. *63*, 101077.

Shorten, C., and Khoshgoftaar, T.M. (2019). A survey on image data augmentation for deep learning. J. Big Data *6*, 60.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Driessche, G.V.D., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., and Lanctot, M. (2016). Mastering the game of Go with deep neural networks and tree search. Nature *529*, 484–489.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In Proceedings of the 31st International Conference on Machine Learning, pp. 1387–1395.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., and Bolton, A. (2017). Mastering the game of go without human knowledge. Nature *550*, 354–359.

Simard, P.Y., Steinkraus, D., and Platt, J.C. (2003). Best practices for convolutional neural networks applied to visual document analysis. In Proceedings of the Seventh International Conference on Document Analysis and Recognition, pp. 958–963.

Sindhwani, V., Melville, P., and Lawrence, R.D. (2009). Uncertainty sampling and transductive experimental design for active dual supervision. In Proceedings of the 26th Annual International Conference on Machine Learning, pp. 953–960.

Sinha, A.P., and Zhao, H. (2008). Incorporating domain knowledge into data mining classifiers: an application in indirect lending. Decis. Support Syst. *46*, 287–299.

Sinha, S., Ebrahimi, S., and Darrell, T. (2019). Variational adversarial active learning. In Proceedings of the IEEE International Conference on Computer Vision, pp. 5972–5981.

Small, K., Wallace, B.C., Brodley, C.E., and Trikalinos, T.A. (2011). The constrained weight space svm: learning with ranked features. In Proceedings of the 28th International Conference on International Conference on Machine Learning, pp. 865–872.

Song, Y., Li, Y., Li, C., and Zhang, G. (2012). An efficient initialization approach of Q-learning for mobile robots. Int. J. Control. Autom. *10*, 166–172.

Speer, R., Chin, J., and Havasi, C. (2016). Conceptnet 5.5: an open multilingual graph of general knowledge. arXiv. arXiv:1612.03975.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. *15*, 1929–1958.

Stewart, R., and Ermon, S. (2017). Label-free supervision of neural networks with physics and domain knowledge. In Proceedings of the 31st AAAI Conference on Artificial Intelligence, pp. 2576–2582.

Su, C., Borsuk, M.E., Andrew, A., and Karagas, M. (2014). Incorporating prior expert knowledge in learning Bayesian networks from genetic epidemiological data. In 2014 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, pp. 1–5.

Su, J. (2018). GAN-QP: a novel GAN framework without gradient vanishing and Lipschitz constraint. arXiv. arXiv:1811.07296.

Su, J., Vargas, D.V., and Sakurai, K. (2019). One pixel attack for fooling deep neural networks. IEEE Trans. Evol. Comput. *23*, 828–841.

Summers, C., and Dinneen, M.J. (2019). Improved mixed-example data augmentation. In n 2019 IEEE Winter Conference on Applications of Computer Vision, pp. 1262–1270.

Sun, J., Fu, Y., and Wan, Q. (2018). Organic synaptic devices for neuromorphic systems. J. Phys. D Appl. Phys. *51*, 314004.

Sutton, R.S., and Barto, A.G. (2018). Reinforcement Learning: An Introduction (MIT press).

Sutton, R.S., McAllester, D.A., Singh, S.P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In Advances in Neural Information Processing Systems, S.A. Solla, T.K. Leen, and K. Müller, eds. (Neural Information Processing Systems Foundation), pp. 1057–1063.

Tercan, H., Guajardo, A., Heinisch, J., Thiele, T., Hopmann, C., and Meisen, T. (2018). Transfer-learning: bridging the gap between real and simulation data for machine learning in injection molding. Proced. CIRP *72*, 185–190.

Thomson, A.M. (2010). Neocortical layer 6, a review. Front. Neuroanat. *4*, 13.

Tian, N., Ji, Z., and Lai, C.-H. (2015). Simultaneous estimation of nonlinear parameters in parabolic partial differential equation using quantum-behaved particle swarm optimization with Gaussian mutation. Int. J. Mach. Learn. Cybern. *6*, 307–318.

Trottier, L., Giguere, P., and Chaib-draa, B. (2017). Multi-task learning by deep collaboration and application in facial landmark detection. arXiv. arXiv:1711.00111.

Tuchman, Y., Mangoma, T.N., Gkoupidenis, P., Burgt, Y.van de, John, R.A., Mathews, N., Shaheen, S.E., Daly, R., Malliaras, G.G., and Salleo, A. (2020). Organic neuromorphic devices: past, present, and future challenges. MRS Bull. *45*, 619–630.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, S. Vishwanathan, and R. Garnett, eds. (Curran Associates), pp. 5998–6008.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K.J. (1989). Phoneme recognition using time-delay neural networks. IEEE Trans. Signal Process. *37*, 328–339.

Wang, J., Gou, L., Shen, H.-W., and Yang, H. (2018). Dqnviz: a visual analytics approach to understand deep Q-networks. IEEE Trans. Vis. Comput. Graph *25*, 288–298.

Wang, Z., Dai, Z., Póczos, B., and Carbonell, J. (2019). Characterizing and avoiding negative transfer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 11293–11302.

Wang, Z., Du, B., Zhang, L., and Zhang, L. (2016). A batch-mode active learning framework by querying discriminative and representative samples for hyperspectral image classification. Neurocomputing *179*, 88–100.

Ware, M., Frank, E., Holmes, G., Hall, M., and Witten, I.H. (2001). Interactive machine learning: letting users build classifiers. Int. J. Hum. Comput. *55*, 281–292.

Watkins, C.J., and Dayan, P. (1992). Q-learning. Mach. Learn. *8*, 279–292.

Weinberger, K.Q., and Saul, L.K. (2006). An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In AAAI Proceedings of the 21st National Conference on Artificial Intelligence, pp. 1683–1686.

Wen, Q., Sun, L., Song, X., Gao, J., Wang, X., and Xu, H. (2020). Time series data augmentation for deep learning: a survey. arXiv. arXiv:2002.12478.

Whitrow, C., Hand, D.J., Juszczak, P., Weston, D., and Adams, N.M. (2009). Transaction aggregation as a strategy for credit card fraud detection. Data Min. Knowl. Discov. *18*, 30–55.

Willett, R., Nowak, R., and Castro, R.M. (2006). Faster rates in regression via active learning. In Advances in Neural Information Processing Systems, Y. Weiss, B. Schölkopf, and J.C. Platt, eds. (MIT Press), pp. 179–186.

Williams, J.D., Asadi, K., and Zweig, G. (2017). Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. arXiv. arXiv:1702.03274.

Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. Chemom. Intell. Lab. Syst. *2*, 37–52.

Wong, S.C., Gatt, A., Stamatescu, V., and McDonnell, M.D. (2016). Understanding data augmentation for classification: when to warp? In 2016 International Conference on Digital Image Computing: Techniques and Applications (IEEE), pp. 1–6.

Worrall, D.E., Garbin, S.J., Turmukhambetov, D., and Brostow, G.J. (2017). Harmonic networks: deep translation and rotation equivariance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5028–5037.

Wu, B., Han, S., Shin, K.G., and Lu, W. (2018). Application of artificial neural networks in design of lithium-ion batteries. J. Power Sourc. *395*, 128–136.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S.Y. (2020). A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. https://doi.org/10.1109/TNNLS.2020.2978386.

Wu, Z., Shen, C., and Hengel, A.V.D. (2019). Wider or deeper: revisiting the ResNet model for visual recognition. Pattern Recognit *90*, 119–133.

Xu, J.-G., Zhao, Y., Chen, J., and Han, C. (2015). A structure learning algorithm for Bayesian network using prior knowledge. J. Comput. Sci. Technol. *30*, 713–724.

Xu, X., Zuo, L., and Huang, Z. (2014). Reinforcement learning algorithms with function approximation: recent advances and applications. Inf. Sci. *261*, 1–31.

Yang, Y., and Loog, M. (2018). A variance maximization criterion for active learning. Pattern Recognit *78*, 358–370.

Yang, Y., Ma, Z., Nie, F., Chang, X., and Hauptmann, A.G. (2015). Multi-class active learning by uncertainty sampling with diversity maximization. Int. J. Comput. Vis. *113*, 113–127.

Ye, C., Yung, N.H., and Wang, D. (2003). A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance. IEEE Trans. Syst. Man. Cybern. Syst. *33*, 17–27.

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. In Advances in Neural Information Processing Systems, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds. (Curran Associates), pp. 4800–4810.

Yuan, H., Paskov, I., Paskov, H., González, A.J., and Leslie, C.S. (2016). Multitask learning improves prediction of cancer drug sensitivity. Sci. Rep. *6*, 31619.

Yun, S., Choi, J., Yoo, Y., Yun, K., and Choi, J.Y. (2017). Action-decision networks for visual tracking with deep reinforcement learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2711–2720.

Zeiler, M.D., and Fergus, R. (2014). Visualizing and understanding convolutional networks. In European Conference on Computer Vision (Springer), pp. 818–833.

Zhang, Q., Wu, Y.N., and Zhu, S.-C. (2018). Interpretable convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8827–8836.

Zhang, R. (2019). Making convolutional networks shift-invariant again. arXiv. arXiv:1904.11486.

Zhang, Z., Kag, A., Sullivan, A., and Saligrama, V. (2019). Equilibrated recurrent neural network: neuronal time-delayed self-feedback improves accuracy and stability. arXiv. arXiv:1903.00755.

Zhang, Z., Luo, P., Loy, C.C., and Tang, X. (2014). Facial landmark detection by deep multi-task learning. In European Conference on Computer Vision (Springer), pp. 94–108.

Zhao, L., Chen, Z., Yang, L.T., Deen, M.J., and Wang, Z.J. (2019). Deep semantic mapping for heterogeneous multimedia transfer learning using co-occurrence data. ACM Trans. Multimedia Comput. Commun. Appl. *15*, 1–21.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. (2020). Random erasing data augmentation. In Proceedings of the 34th AAAI Conference on Artificial Intelligence, pp. 13001–13008.

Zhou, J.T., Pan, S.J., and Tsang, I.W. (2019). A deep learning framework for hybrid heterogeneous transfer learning. Artif. Intell. *275*, 310–328.

Zhou, J.T., Pan, S.J., Tsang, I.W., and Yan, Y. (2014). Hybrid heterogeneous transfer learning through deep learning. In Proceedings of the 38th AAAI Conference on Artificial Intelligence, pp. 2213–2219.

Zhou, S., Helwa, M.K., and Schoellig, A.P. (2017). Design of deep neural networks as add-on blocks for improving impromptu trajectory tracking. In IEEE 56th Annual Conference on Decision and Control, pp. 5201–5207.

Zou, Z., Shi, Z., Guo, Y., and Ye, J. (2019). Object detection in 20 years: a survey. arXiv. arXiv:1905.05055.