MINI-REVIEW

Dealing with Multi-Dimensional Data and the Burden of Annotation

Easing the Burden of Annotation

Benjamin R. Mitchell,* Marion C. Cohen,† and Stanley Cohen‡

Artificial Intelligence 101

The need for huge data sets represents a bottleneck for the application of artificial intelligence. Substantially fewer annotated target lesions than normal tissues for comparison present an additional problem in the field of pathology. Organic brains overcome these limitations by utilizing large numbers of specialized neural nets arranged in both linear and parallel fashion, with each solving a restricted classification problem. They rely on local Hebbian error corrections as compared to the nonlocal back-propagation used in most artificial neural nets, and leverage reinforcement. For these reasons, even toddlers are able to classify objects after only a few examples. Rather than provide an overview of current AI research in pathology, this review focuses on general strategies for overcoming the data bottleneck. These include transfer learning, zero-shot learning, Siamese networks, one-class models, generative networks, and reinforcement learning. Neither an extensive mathematic background nor advanced programing skills are needed to make these subjects accessible to pathologists. However, some familiarity with the basic principles of deep learning, briefly reviewed here, is expected to be useful in understanding both the current limitations of machine learning and determining ways to address them. (Am J Pathol 2021, 191: 1709–1716; https://doi.org/10.1016/j.ajpath.2021.05.023)
nearest neighbor), probabilistic (Bayesian), or stratification (random forest) strategies, among others. One of the most powerful varieties of the former is the support vector machine, which involves finding a decision boundary between different classes in the data (reviewed by Cohen1).

All of these models utilize training sets consisting of known, labeled examples. Each sample is defined in terms of a finite set of attributes (features), and its attribute set is known as a feature vector. The feature vector can be represented as a point in \( n \)-dimensional space, where \( n \) is the number of attributes needed to classify the sample. For example, the classes of caterpillars and ladybugs can be distinguished by using two attributes, length and width, so that the feature vector is two-dimensional. In this case, the decision boundary is a line. For an \( n \)-dimensional feature vector, the boundary is a hyperplane, and it has \( n - 1 \) dimensions.

NNs also make use of data in the form of feature vectors, but do not do so by explicitly calculating hyperplanes. Unlike a support vector machine, the operation of a NN is (very loosely) based on an analogy with a biological neuron that receives multiple inputs from other neurons, and if the sum of those inputs exceeds a threshold value, the neuron sends a signal to the next neuron. In a basic NN, the nodes (neurons) are arranged in layers, with every neuron in one layer being directly connected to all of the neurons in the next. Each input is given a weight, and initially these are randomly assigned. The weights are modified (via a process known as error back-propagation,\(^4\) based on a training set of known examples, until the model becomes proficient in predicting the class of an unknown sample it has never seen before. By virtue of the multiple interconnections from one layer to the next, each layer provides an increasingly complex representation of the feature vector.

Although all of these approaches to machine learning remain important, in recent years NNs have come to dominate the field of AI because of their power and flexibility. NNs are often referred to as examples of deep learning because of their multilayer architecture. Although it seems that NNs represent a new evolution of machine learning, their roots were in the mid-20th century, at approximately the same time as the simpler kinds of machine learning were being used. It was the limitations of early NNs that delayed their adoption until innovative research and advances in computer hardware allowed NNs to evolve to overcome those limitations.

For megapixel images (as well as many other kinds of large data arrays), the basic NNs must be modified so as to allow the early layers of the NNs to find representations of the data that are much smaller than the original number of input features, and the NN architecture that does this best for images is known as a convolutional neural network.\(^5\) Most, but not all, modern AI is based on deep learning. A detailed explanation of the basics of machine learning and examples of their use in pathology, with emphasis on deep learning, can be found in a chapter by Cohen.\(^1\)

### Humans in the Loop: Harvesting Usable Data

As indicated above, in all of these machine-learning algorithms, the computer learns best from a set of labeled (annotated) examples (samples) that fall into several classes. Based on this training, it can then make predictions about unknown samples it has not previously seen. Since the computer learns by example, this is known as supervised learning. Note that as with all AI, this is learning in the sense of improving performance through experience and is not intelligent behavior as we understand it in humans.

Because they are much more powerful, NNs have largely, but not completely, replaced the other forms of machine learning in fields such as pathology. However, given that most NNs are very complex and data in clinical practice have large numbers of attributes, they require vast amounts of known data for training. Large data sets for pathology are very hard to come by other than in some major medical centers, and they require a great deal of human input for proper labeling (annotation). There is also the problem of class imbalance, as for example in tumors, where the set of samples of specific tumor types is often smaller than the set of samples of corresponding normal tissue. Additionally, NNs trained on material from one center may not perform as well on data from another center. Another problem is that all of these algorithms are one-trick ponies; it is hard to train a model to handle multiple tasks, including even related ones. The computer basically suffers from savant syndrome: it can do one task incredibly well but does poorly in everything else.

These limitations represent a major challenge for moving forward in the use of AI in pathology. A hint as to how this challenge can be overcome can be found in the observation that computer learning as described above has very little to do with the learning that biological brains are capable of. For example, it does not take more than a few bananas to teach a toddler what a banana is, or the difference between a guitar and a radio, even though they both play music. Pigeons can discriminate between cancer and normal tissue based on as few as 200 examples\(^6\) by being rewarded for correct responses (ie, reinforcement learning).

The brain uses what is known as Hebbian learning,\(^7\) in which changes in synaptic strength depend only on activity across the local connections to and from a neuron (colloquially, neurons that fire together, wire together). Thus, biological learning is essentially local. Nonlocality is an emergent process that arises globally through interconnections involving many such local networks distributed throughout the brain. In contrast, current models of AI are intrinsically nonlocal, in that the weight updates depend on all of the neural layers, working backward from output to input. It is likely that models that process data like a brain may reduce the amount of training data to that required for a human, or even a pigeon. Unfortunately, such models are not yet tractable for large-scale problems such as...
those routinely encountered in pathology. Until such new paradigms are fully developed, one can modify current NNs in a variety of ways by relaxing dependence on strictly supervised learning. This review provides an overview of some of these approaches.

Reducing the Need for Annotated Data

It would be nice for a computer to learn in a completely unsupervised manner, without annotated training input from a human. However, even humans do not learn in this fashion. While there are algorithms for unsupervised learning, the usual approach is to combine some unsupervised steps with some supervised ones (semi-supervised learning). For sparsely supervised learning, only a small number of labeled, annotated examples are available. For weakly supervised learning, there may be only partial annotation; for example, one may know only whether a tissue sample contains cancer, but nothing about the distribution or extent of the cancer within the sample, or about local differences in its morphology. An alternative approach, called transfer learning, is to take advantage of previously trained models from other domains similar to the one that is being studied. This review discusses mainly sparsely supervised learning; weakly supervised learning utilizes similar techniques in conjunction with segmentation algorithms. In all cases, semi-supervised learning begins with an unsupervised strategy.

Overview of Unsupervised Pretraining

The basic idea behind the use of unsupervised pretraining is that the unsupervised step is essentially finding a feature-based representation of the data suitable for reconstructing the original inputs without knowing their class labels.

These representations are then used as starting points for subsequent supervised training. As long as a similar set of features is used for classification, pretraining the network in an unsupervised setting can allow the supervised learning stage to achieve good convergence, even in cases in which the amount of labeled data would otherwise be too small to allow it.

It should be noted that unsupervised pretraining is not a magic bullet; in particular, it can reduce the amount of labeled data required to a point, but there will still be a point at which there are simply too few labeled data to learn a model that generalizes well. Additionally, in nearly all cases, unsupervised pretraining is outperformed by fully supervised training using the same amount of data (ie, if you have labels for all of your data, you are better off simply doing supervised training from the beginning). That said, this method is still potentially useful, particularly in cases in which the acquisition of large amounts of labeled data is considerably more expensive than is the acquisition of large amounts of unlabeled data.

Transfer Learning

One of the simplest ways of dealing with data that has a small number of labeled examples is to piggyback them onto a model that has been trained on other data sets that are similar but not related.\textsuperscript{9} Recall that the NN builds up a series of more complex representations of the data from layer to layer. The intuition behind transfer learning is that if two problems involve similar kinds of data, then the low-level features of the earlier representations are likely to be similar. If this is true, instead of beginning with random weightings of the inputs in the early layers of the new NN, the weightings that were obtained in training the previous model can be used.

Transfer learning involves starting with a NN trained by someone else. There are a number of pretrained networks available in publicly accessible deep-learning toolkits, and pretrained NNs are becoming increasingly available to pathologists. The more closely the data sets of the pretrained and the to-be-trained models match, the more effective this strategy will be. For example, if one were training a NN to distinguish benign from malignant prostate neoplasms, it would be better to start with a known model trained on breast tissue for this purpose, rather than a model that has been trained to distinguish roses from orchids.

The actual transfer learning process involves taking the network that is already trained and removing the fully connected higher level(s) from the network while keeping the weightings of the earlier layers frozen. The upper layer(s) are replaced with a new, fully connected upper level with randomly initialized weights, as the resulting hybrid is trained using the new data set. The primary advantage is that the training can succeed with a much smaller labeled training set than would be required to train the same network from scratch (ie, starting with randomly initialized weights). Transfer learning is thus an important part of the digital pathologist’s armamentarium.

Unsupervised Pretraining via Clustering

Geometric cluster-based algorithms can be used as purely unsupervised learning for the early stages of a more complex model. The simplest such model is the $K$-means (KM) algorithm.\textsuperscript{9} KM attempts to find clustering in unlabeled data. It starts by choosing a value of $K$ (the number of clusters) based on the number of classes to be distinguished.

Next, a random guess is made as to where the central points of each cluster are so as to generate $K$ pseudo-centers. These pseudo-centers should be reasonably distant from each other but need have no relationship to the actual clusters and can be chosen randomly (unless there is some prior knowledge about the data structure). Each point of the
data set is then assigned to its nearest pseudo-center. This forms clusters, with each cluster comprising all of the data points associated with its pseudo-center. Next, the location of each cluster’s pseudo-center is updated so that it is in the center of all of its members. This is done repeatedly until there are no more changes to cluster membership. This generates \( K \) clusters, each consisting of a group mainly, if not exclusively, of one class. It is necessary to label only a small number of samples from each cluster and assign those labels to every member of that cluster. In this manner, one can generate a large data set without the need for manual annotation of every sample of that data set.

This way sufficient training data are generated for a NN. Clustering is unsupervised; the resulting NN is sparsely supervised.

A more sophisticated use of clustering is a method known as \( N \)-shot or few-shot learning.\(^{10} \) A shot is defined as a single example available for training. In \( N \)-shot learning, there are only a few examples of each class (for one or a small number of examples, it is often called one-shot learning). The task is to classify any test image to a class using that constraint. One way of doing this is by the use of prototypical networks.\(^{11} \)

Unlike the usual deep-learning architecture, prototypical networks do not classify the images directly. Instead, they learn the mapping of the image into metric space, meaning that images of the same class are placed in proximity to each other, while different classes are placed at further differences. This, like \( K \)-means, is a geometric model.

However, instead of using a basic \( K \)-means as a preprocessing step, the whole structure is built into a convolutional NN. Algorithms in this category are often known as Image2Vector models. Even more sophisticated models involve NNs augmented with a neural attention mechanism; a good example of a disease-related application of few-shot learning to image classification is available in the article by Chen et al.\(^{12} \)

One-Class Learning

It is important to distinguish between one-shot learning and one-class learning. The object of one-class classification, alias anomaly detection, is to recognize instances of a class by using only examples of that class.\(^{13} \) All other classes are lumped together and are known as alien classes. In other words, the difference between one-class and binary classification is the absence of training data from a second class. While this approach has received a great deal of attention for simple data sets, it has proven difficult to obtain good results in data sets from clinical practice, such as those encountered in pathology. However, this is probably more like human learning than the large data set–dependent multiclass models, so it remains a topic worthy of investigation. Recently, Perrera and Patel\(^{14} \) and Sokolov et al\(^{15} \) described one-class learning for anomaly (outlier) detection. More directly related to pathology, this strategy has been used to distinguish tumor transcriptomes as compared to normal in this manner.\(^{16} \) Such approaches could, in principle, detect foci of cancer in otherwise normal tissue, by considering those foci as anomalies; in other words, based on a training protocol in which normal tissue images are used and a metastatic focus is then treated as an anomalous outlier. As of the date of the preparation of this article, no examples of work in this area were found. However, industrial techniques involving the detection of anomalies in images of surfaces suggest that this is a feasible avenue to explore.

A similar strategy involves the use of Siamese networks.\(^{17,18} \) Instead of a model that learns to classify its inputs like the usual NN, the NN learns to differentiate between two inputs. A typical Siamese network consists of two identical NNs, each taking one of the two images. Identical means that they have the same configuration with the same parameters and weights. The last layers of the two networks are jointly fed into a contrastive loss function that calculates the degree of similarity between the two input images. An example from natural language processing involving scoring an examination question may make this approach more intuitive. Here, one input is the question sentence, and the other input is the answer. The output is how relevant the answer is to the question. This is not a simple supervised learning task, in that there is no database consisting of all possible answers that are judged correct and examples that are judged wrong. Nonetheless, a Siamese network can extract similarity or a connection between question and answer. In the article by Cano and Cruz-Roa,\(^{18} \) this approach was used for image classification in breast cancer. In addition to detecting similarities and differences in pairs such as cancer versus noncancer, Siamese networks can also be used to analyze patterns within image patches from whole slide images, and thus may be useful for segmentation. For example, Gildenblat and Klaiman\(^{19} \) used a sophisticated model for image retrieval from patch networks using a Siamese network consisting of two branches of a modified residual NN (ResNet-50) pretrained on the ImageNet data set for retrieving areas of tumor from scanned images.

Unsupervised Pretraining via Autoencoders

There are other strategies for discovering useful representations without class labels that are based on generative models. A classic example is to first train one or more layers of a NN as an autoencoder.\(^{20} \) The autoencoder is trained simply to reconstruct its input; that is, the desired output is merely a copy of the input. If the network has enough hidden nodes, this task becomes trivial, but if the network is restricted in capacity, then it is forced to learn a low-dimensional representation of the input data. This can be thought of as analogous to a lossy compression scheme; the
compressed data are encoded by the activations of the hidden nodes.

Once the unsupervised model has been trained, it is then used as the basis for a supervised model, typically by simply adding one or two extra, fully connected layers to the top. At that point, the entire network is trained using a standard supervised approach. This results in more efficient training of a NN and may in some cases reduce the number of annotated examples needed for training.

If an even more abstract and compressed representation is desired, the output of one autoencoder layer can be used as the input for another. The result is called a stacked autoencoder. Other variations of autoencoders, such as variational autoencoders and sparse autoencoders, are also available. Autoencoders are not only useful in classification, they can also be used for tasks such as stain normalization as a preprocessing step for both human and computer image interpretation.\textsuperscript{21}

### Unsupervised Pretraining via Generative Adversarial Networks

The type of network discussed so far is used for classification or similarity discrimination. While this type of networks are by far the most popular type, they are not the only type used. There are several other types of problems that can be tackled using machine learning, and one of them involves generative modeling. A generative model is one that can be used to generate new examples. The basic task is to take in a set of examples, and then construct new examples that are unique, but are similar to the examples the system was trained on. In statistical terms, the goals are to model the distribution that the training samples were drawn from, and then to generate new samples drawn from the same distribution.

There are several cases in which it might be worthwhile to train such a generative model. First, one might directly want the outputs; that is, the generation of plausible synthetic examples may be useful in its own right. Second, one may view the generative model as another way of finding potentially useful features, which could then be exploited as a method of unsupervised pretraining (similar to the use of autoencoders). Third, one may be interested in a conditional generative model, which is a fancy way of saying that one is interested in transforming inputs, rather than generating them from whole cloth (for instance, taking an image and generating a version in the style of some famous artist). All three of these things are possible using generative models.

In the context of deep NNs, a powerful generative model is the generative adversarial network.\textsuperscript{22} Generative adversarial networks work by generating two networks and training them in competition with one another. The first network is called the \textit{generator}, and it is involved in generating plausible synthetic (fake) examples. The second network is called the \textit{discriminator}, and it is involved in distinguishing between real examples (taken from the training set), and fake examples (generated by the generator). These two networks are effectively playing a zero-sum game, in which each player’s success must come at the expense of the other. The way for the discriminator to win is to figure out how to recognize the generator’s artificial examples, and the way for the generator to win is to make better fakes. After many training iterations, the fakes become indistinguishable from the real thing. At that point, the generator will have generated a compact internal representation of the original. The discriminator can then be replaced with a conventional neural net and proceed with training. This is analogous to using early internal representations from another model as the starting point for the new model, as was the case for transfer learning. In both cases, supervised learning starts with informed guesses about the weights of the earlier layers, rather than random values, so one can get away with less labeled examples. Another use would be to tweak the representations generated by the generator so as to add a bit of variance to them, and then use them as additional labeled examples for training. This is one way to generate synthetic data. By expanding the labeled data set in this manner, one can start with a smaller set of labeled data than would otherwise be required.

Generative adversarial networks can do much more than this, but a deeper dive is beyond the scope of this article. However, Tschuchnig et al.\textsuperscript{23} recently published an excellent review of the use of generative adversarial networks in digital pathology.

### Reinforcement Learning

To reiterate, most NN models are trained in a supervised fashion, such that the system is given a labeled training set consisting of hand-labeled input-output pairs. However, this is not practical for certain tasks. In particular, for sequential decision problems, the performance of the system can often be accurately assessed only at certain points. A common example is trying to train a network to play a game like Chess or Go; until the game is over, it is unknown whether the strategy being used is a good one. The network cannot be trained to decide what moves to make in the middle of the game using standard supervised learning.

The idea of reinforcement learning (RL) was developed for such situations. Inspired by operant conditioning, RL works by providing a reward signal that tells the system when it is doing well and when it is doing poorly. Over time, the system experiments and, through trial and error, discovers a policy that will maximize its expected reward.

The classic formulations of this algorithm are based on the formalism of a Markov decision process, which encodes a problem as a discrete set of possible states and actions, along with a transition model that specifies a probability
distribution over possible next states given a current state and choice of action by an agent.

Given a reward function that specifies which reward the agent receives for being in any given state, it is possible to use an iterative approximation algorithm to determine the optimal action to take in any state, where optimal is defined as the action that leads to the highest expected future reward. It is also common to add a discount factor, which allows for down-weighting of future expected gain based on how distant it is; by tuning this parameter, the agent can be made to optimize for either more short-term or more long-term reward. It is important to stress that the agent is not given a set of defined paths as a training set; in this sense, RL is a kind of unsupervised learning.

Recently, RL has been used for solving classification problems. Here, instead of a series of actions leading to a goal such as avoiding obstacles, the agent performs a classification action at one sample, and the classification action is evaluated and returns a reward to the agent. This turns out to be useful in cases of imbalanced data distributions, a common situation in pathology, in which there are usually many more examples of normal than of diseased tissue samples. Lin et al. did so by simply making the reward from the minority class sample larger than the reward from the majority class sample. They showed that this approach outperforms more conventional approaches dealing with imbalance. The primary downside of RL is its lack of computational efficiency; training via RL is very slow compared to fully supervised learning. Furuta et al. recently described PixelRL, an algorithm that combines RL with a fully convolutional network. Hybrid algorithms such as these hold great promise for future advances in AI.

Self-Supervised Learning

All of the approaches described in this review have a major limitation of AI as it exists today. While they are trained to do tasks that do not require active thinking, they are not designed for the task at which humans excel, namely, reasoning. The challenge is to generate deep-learning systems that can learn and plan complex action sequences and decompose tasks into subtasks. In other words, deep-learning systems are good at providing end-to-end solutions but bad at breaking them down into specific interpretable and modifiable steps that are the building blocks of reasoning about complex tasks, and lever previously acquired knowledge. There are some baby steps in this direction, such as capsule networks and compositional learning. Lecun, the father of convolutional networks, took what are arguably the first systematic steps toward the development of self-supervising machines through an approach that he defined as variable energy-based models (EBMs). The approach is to learn an energy function that takes low values on the data manifold and higher values everywhere else through mappings based on hidden variables. To quote from this reference, "Energy-based models (EBMs) capture dependencies between variables by associating a scalar energy to each configuration of the variables. Inference consists in clamping the value of observed variables and finding configurations of the remaining variables that minimize the energy. Learning consists in finding an energy function in which observed configurations of the variables are given lower energies than unobserved ones. The EBM approach provides a common theoretical framework for many learning models, including traditional discriminative and generative approaches."

EBM allows for the initial stages of training to be accomplished by the use of large amounts of unlabeled data for initial training and looking for relevant associations. For example, in analyzing a video stream, an EBM can predict a future frame from the current one (as well as a past frame from the current one). It can also infer missing parts of a sample. EBMs can be used for classification tasks. In a typical neural architecture, the image is fed to a NN. The dependencies at the granular level are captured and given some probability scores. However, in the case of EBMs, classification is based solely on energy values. This approach significantly reduces the number of labeled, annotated samples necessary.

Quantum Computing to the Rescue

This review focused on methods of dealing with situations in which annotated data are limited. There is another potential problem. What happens when annotated data become overwhelming? Humans collectively produce an estimated 2.5 exabytes of data per day. The limits of the data-processing power of conventional computers have been reached, and data keep growing faster than improvements in their circuitry. It is therefore now possible to pose problems that because of scale cannot be resolved in timescales that are reasonable by human standards. Fortunately, computer architectures that take advantage of quantum effects such as super-positioning and entanglement allow for a massive parallelism that can enable solution of such problems in seconds, rather than many years. These algorithms were originally simulated on conventional computing devices to determine feasibility, but have now been realized in actual quantum devices.

There are a number of ways to generate detectable quantum states. One example is to use a device manufactured by D-Wave Systems (Burnaby, BC, Canada) that is based on superconducting electrical loops. These loops are wired together to allow them to interact magnetically. Many other quantum circuit topologies have been implemented. In addition to D-Wave, IBM (Armonk, NY), Microsoft (Redmond, WA), Google (Mountain View, CA), and others have functioning quantum computers. Some of these are available on the cloud for general access by the public.
Although the mechanisms of quantum computation are beyond the scope of this discussion, it suffices to say that a major difference between conventional and quantum computing lies in the computational building blocks used. While a conventional computer uses bits, which can take on the values of either zero or one, the qubit is neither; it exists in an indeterminate state that is neither zero nor one until it is measured. Also, given that particles are also waves, qubits are associated with a phase value. Thus, a qubit can be said to exist in a large number of possible states (superposition) until it is measured, at which point it collapses to a single value. The quantum algorithm is a strategy that, for a given input, causes an array of qubits to collapse into the correct configuration (solution).

AI for the most part requires annotated data, which is a small subset of the total amount of existing data. However, for images alone, there are a number of centers with millions of annotated digitized slides, and in short order, there will be another 50 million from but one single additional source. To abstract meaningful patterns from data of this magnitude will require quantum computation, and this has led to the hybrid field of quantum artificial information. One big advantage of quantum computing applied to AI is the number of dimensions it can process. In the time that a classic artificial neuron can process an input of \( N \) dimensions, a quantum perceptron can process \( 2^N \) dimensions.30

An early application of quantum computing, the Grover algorithm, was designed to identify a unique sample in a large, unsorted database. A classic computer can do this kind of sorting problem for \( N \) samples, in \( N \) steps, but the Grover algorithm requires only the square root of \( N \). The Grover algorithm has an obvious analogy with the problem of pattern recognition or image classification, and recently, it has been applied to that task as well. There are currently analogues of classic perceptrons and Hopfield networks, and quantum learning has been applied to unsupervised learning and RL as well. The integration of quantum computing with NNs is described in detail by Wittek31 as well as other examples of machine learning. Ultimately, quantum artificial information will be able to integrate very different data sets (eg, transcriptomes and morphology), which is difficult without human intervention at present.

Interestingly, the human brain, with its massive parallelism and capacity to process multiple computational tasks simultaneously, seems to function a bit like a quantum computer, although it does so at room temperature rather than at the ultralow temperatures that are currently required to retain coherent states. Quantum effects may also play a role in other biological and biochemical processes such as photosynthesis (reviewed by Cao et al32).

**Summary**

Rather than provide an overview of current research articles related to AI in the literature on pathology, this review focused on the strategies for dealing with the huge data sets that are routinely encountered. Much of these data may be either unlabeled, or in situations in which segmentation is needed, incompletely annotated.

Although not explicitly stated, the discussion was mainly image focused (pun intended), although the strategies, in broad outline, hold for other tasks such as natural language processing, risk prediction, and so on. For simplicity, but without loss of generality, the discussion was based on simple convolutional NNs, but in clinical practice, more advanced architectures, such as residual networks, recurrent NNs, and deep belief systems, have been used with success. This review outlined and discussed various strategies for reducing the number of annotated samples necessary for training AI. Even as this area is improved, it is highly unlikely that AI will supplant human pathologists, for many reasons. One such reason is related to the differences between human errors and machine errors. Utilizing approaches described here and in the more specific examples of the subsequent reviews in this issue, it seems that humans and computers make approximately the same number and kinds of mistakes. What is interesting is that they make different mistakes. Therefore, it is likely that a partnership between human and machine will be better than either one alone. For example, one can envision interactions between man and machine for collaborative annotation and for improvements in the assignment of ground truth labels based on such interactions. In the lay press, this kind of man–machine hybrid has been described as a centaur. With this collaboration, AI will evolve from being a pathologist’s digital assistant to being a (nontenured) pathologist’s partner and colleague rather than a replacement for the pathologist. The rapidly expanding field of quantum AI, although still in the embryonic stage, is expected to accelerate the evolution of this partnership.

**References**