Does Machine Learning Really Work?

Tom M. Mitchell

■ Does machine learning really work? Yes. Over the past decade, machine learning has evolved from a field of laboratory demonstrations to a field of significant commercial value. Machine-learning algorithms have now learned to detect credit card fraud by mining data on past transactions, learned to steer vehicles driving autonomously on public highways at 70 miles an hour, and learned the reading interests of many individuals to assemble personally customized electronic newspapers. A new computational theory of learning is beginning to shed light on fundamental issues, such as the trade-off among the number of training examples available, the number of hypotheses considered, and the likely accuracy of the learned hypothesis. Newer research is beginning to explore issues such as long-term learning of new representations, the integration of Bayesian inference and induction, and life-long cumulative learning. This article, based on the keynote talk presented at the Thirteenth National Conference on Artificial Intelligence, samples a number of recent accomplishments in machine learning and looks at where the field might be headed.

Ever since computers were invented, it has been natural to wonder whether they might be made to learn. Imagine computers learning from medical records to discover emerging trends in the spread and treatment of new diseases, houses learning from experience to optimize energy costs based on the particular usage patterns of their occupants, or personal software assistants learning the evolving interests of their users to highlight especially relevant stories from the online morning newspaper. A successful understanding of how to make computers learn would open up many new uses for computers. And a detailed understanding of information-processing algorithms for machine learning might lead to a better understanding of human learning abilities (and disabilities) as well.

Although we do not yet know how to make computers learn nearly as well as people learn, in recent years, many successful machine-learning applications have been developed, ranging from data-mining programs that learn to detect fraudulent credit card transactions to information-filtering systems that learn users' reading preferences to autonomous vehicles that learn to drive on public highways. Along with this growing list of applications, there have been important advances in the theory and algorithms that form the foundations of this field.

The Niche for Machine Learning

One way to understand machine learning is to consider its role within the more broad field of computer science. Given the broad range of software paradigms within computer science, what is the niche to be filled by software that learns from experience? Obviously, for computer applications such as matrix multiplication, machine learning is neither necessary nor desirable. However, for other types of problems, machine-learning methods are already emerging as the software development method of choice. In particular, machine learning is beginning to play an essential role within the following three niches in the software world: (1) data mining, (2) difficult-to-program applications, and (3) customized software applications.

Data mining: *Data mining* is the process of using historical databases to improve subsequent decision making. For example, banks now analyze historical data to help decide which future loan applicants are likely to be credit worthy. Hospitals now analyze historical

Data:

Patient103 time = 1 →	*Patient103* time = 2 … →	*Patient103* time = *n*
Age: 23	Age: 23	Age: 23
FirstPregancy: no	FirstPregancy: no	FirstPregancy: no
Anemia: no	Anemia: no	Anemia: no
Diabetes: no	Diabetes: YES	Diabetes: no
PreviousPrematureBirth: no	PreviousPrematureBirth: no	PreviousPrematureBirth: no
Ultrasound: ?	Ultrasound: abnormal	Ultrasound: ?
Elective C-Section: ?	Elective C-Section: no	Elective C-Section: no
Emergency C-Section: ?	Emergency C-Section: ?	**Emergency C-Section: Yes**
…	…	…

Learned Rule:

If No previous vaginal deliveray and
Abnormal 2nd Trimester Ultrasound, and
Malpresentation at admissions, and
Then Probability of Emrgency C-Section is 0.6

Training set accuracy: 26/41 - .63
Test set accuracy: 12/20 - .60

Figure 1. A Typical Data-Mining Application.

A historical set of 9714 medical records describes pregnant women over time. The top portion of the figure illustrates a typical patient record, where ? represents that the feature value is unknown. The task here is to identify classes of patients at high risk of receiving an emergency Cesarean section (C-section). The bottom portion of the figure shows one of many rules discovered from these data using Schenley Park Research's RULER system. Whereas 7 percent of all pregnant women received emergency C-sections, this rule identifies a subclass of high-risk patients, 60 percent of whom received emergency C-sections.

In particular, machine learning is beginning to play an essential role within three niches in the software world: data mining, difficult-to-program applications, and customized software applications.

data to help decide which new patients are likely to respond best to which treatments. As the volume of online data grows each year, this niche for machine learning is bound to grow in importance. See www.kdnuggets.com/ for a variety of information on this topic.

Difficult-to-program applications: Machine-learning algorithms can play an essential role in applications that have proven too difficult for traditional manual programming—applications such as face recognition and speech understanding. The most accurate current programs for face recognition, for example, were developed using training examples of face images together with machine-learning algorithms. In a variety of applications where complex sensor data must be interpreted, machine-learning algorithms are already the method of choice for developing software.

Customized software applications: In many computer applications, such as online news browsers and personal calendars, one would like a system that automatically customizes to the needs of individual users after it has been fielded. For example, we would like a news browser that customizes to the individual user's reading interests or an online calendar that customizes to the individual user's scheduling preferences. Because it is unrealistic to manually develop a separate system for each user, machine learning offers an attractive option for enabling software to automatically customize itself to individual users.

The following sections present examples of successful applications of machine-learning algorithms within each of these three niches. As in any short summary of a large field, the role of this article is to sample a few of the many research projects and research directions in the field to provide a feel for the current state of the art and current research issues. More lengthy treatments of the current state of machine learning can be found in Dietterich's article (also in this issue) and in Mitchell (1997). Treatments of various subfields of machine learning are also available for topics such as statistical learning methods (Bishop 1996), neural networks (Chauvin and Rumelhart 1995; Hecht-Nielsen 1990), instance-based learning (Aha, Kibler, and Albert 1991), genetic algorithms (Mitchell 1996; Forrest 1993); computational learning theory (Kearns and Vazirani 1994), decision tree learning (Quinlan 1993), inductive logic programming (Lavraĉ and Dẑeroski 1994), and reinforcement learning (Kaelbling, Littman, and Moore 1996; Barto, Bradtke, and Singh 1995).

Data Mining: Predicting Medical Outcomes from Historical Data

Data mining involves applying machine-learning methods to historical data to improve future decisions. One prototypical example is illustrated in figure 1. In this example, historical data describing 9714 different pregnant women have been provided, with each woman described by a set of 215 attributes over time. These attributes describe the woman's health history (for example, number of previous pregnancies), measurements taken at various times during pregnancy (for example, ultrasound results), the type of delivery (for example, normal, elective Cesarean sections [C-sections], emergency C-section), and final health of mother and baby.

Given such time-series data, one is often interested in learning to predict features that occur late in the time series based on features that are known earlier. For example, given the pregnancy data, one interesting problem is to predict which future patients are at exceptionally high risk of requiring an emergency C-section. This problem is clinically significant because most physicians agree that if they know in advance that a particular patient is at high risk of an emergency C-section, they will seriously consider an elective C-section instead to improve the expected clinical outcome for the mother and child. The bottom half of figure 1 provides one typical rule learned from these historical data. Whereas the risk of emergency C-section for the entire population of patients in this case was 7 percent, this rule identifies a class of patients for which the risk is 60 percent. The rule was learned by the RULER program developed by Schenley Park Research, Inc. The program used two-thirds of the available data as a training set, holding the remaining one-third as a test set to verify rule accuracy. As summarized in the figure, a total of 41 patients in the training set match the rule conditions, of which 26 received an emergency C-section. A similar accuracy is found over the test set (that is, of the 20 patients in the test set who satisfy the rule preconditions, 12 received emergency C-sections). Thus, this rule successfully identifies a class of high-risk patients, both over the training data and over a separate set of test data.

This medical example illustrates a prototypical use of machine-learning algorithms for data mining. In this example, a set of time-series data was used to learn rules that predict later features in the series from earlier features. Time-series prediction problems such as this occur in many problem domains and data sets. For example, many banks have time-series data indicating which credit card transactions were later found to be fraudulent. Many businesses have customer databases that indicate which customers were later found to purchase certain items. Universities have data on which students were later found to successfully graduate. Manufacturers have time-series data on which process parameters later produced flawed or optimal products. Figure 2 illustrates a number of time-series prediction problems with the same abstract structure as the pregnancy example. In all these cases, the problem of discovering predictive relations between earlier and later features can be of great value.

What is the current state of the art for such problems? The answer is that we now have a robust set of first-generation machine-learning algorithms that are appropriate for many problems of this form. These include decision tree–learning algorithms such as C4.5 (Quinlan 1993), rule-learning algorithms such as CN2 (Clark and Niblett 1989) and FOIL (Quinlan 1990), and neural network–learning algorithms such as back propagation (Rummelhart and McClelland 1986). Commercial implementations of many of these algorithms are now available. At the same time, researchers are actively pursuing new algorithms that learn more accurately than these first-generation algorithms and that can use types of data that go beyond the current commercial state of the art. To give a typical example of this style of research, Caruana, Baluja, and Mitchell (1995) describe a multitask learning algorithm that achieves higher prediction accuracy than these other methods. The key to their success is that their system is trained to simultaneously predict multiple features rather than just the feature of interest. For example, their system achieves improved accuracy for predicting mortality risk in pneumonia patients by simultaneously predicting various lab results for the patient.

Data mining involves applying machine-learning methods to historical data to improve future decisions.

What are good topics for new thesis research in this area? There are many, such as developing algorithms for active experimentation to acquire new data, integrating learning algorithms with interactive data visualization, and learning across multiple databases. Two of my current favorite research topics in this area are (1) learning from mixed-media data and (2) learning using private data plus internet-accessible data.

Learning from mixed-media data: Whereas rule-learning methods such as the one discussed in figure 1 can be applied to data

Customer Purchase Behavior

Customer103: (time = t0)	*Customer103:* (time = *t*1) …	*Customer103:* (time = *tn*)
Sex: M	Sex: M	Sex: M
Age: 53	Age: 53	Age: 53
Income: $50k	Income: $50k	Income: $50k
Own House: Yes	Own House: Yes	Own House: Yes
MS Products: Word	MS Products: Word	MS Products: Word
Computer 386 PC	Computer Pentium	Computer Pentium
Purchase Excel?: ?	Purchase Excel?: ?	**Purchase Excel?: Yes**
…	…	…

Customer Retention:

Customer 103: (time = *t*0)	*Customer103:* (time = *t*1)	*Customer103:* (time = *tn*)
Sex: M	Sex: M	Sex: M
Age: 53	Age: 53	Age: 53
Income: $50K	Income: $50K	Income: $50K
Own House: Yes	Own House: Yes	Own House: Yes
Checking: $5k	Checking: $20k	Checking: $0
Savings: $15k	Savings: $0	Savings: $0
Current-customer?: yes	Current-customer?: yes	**Current-customer?: No**

Process Optimization

Product72: (time = *t*0)	*Product72:* (time = *t*1)	*Product72:* (time = *tn*)
Stage: mix	Stage: cook	Stage: cool
Mixing-speed: 60 rpm	Temperature: 325	Fan-speed: medium
Viscosity: 1.3	Viscosity: 3.2	Viscosity: 1.3
Fat content: 15%	Fat content: 12%	Fat content: 12%
Density: 2.8	Density: 1.1	Density: 1.2
Spectral peak: 2800	Spectral peak: 3200	Spectral peak: 3100
Product underweight?: ??	Product underweight?: ??	**Product underweight?: Yes**
…	…	…

Figure 2. Additional Examples of Time-Series Prediction Problems.

described by numeric features (for example, age) and symbolic features (for example, gender), current medical records include a rich mixture of additional media, such as images (for example, sonogram images, x-rays), signals from other instruments (for example, EKG signals), text (for example, notes made on the patient chart), and even speech (for example, in some hospitals, physicians speak into handheld recorders as they make hospital rounds). Although we have machine-learning methods that can learn over numeric and symbolic data, other methods that can learn over images, and other methods that can aid learning with speech data, we have no principled methods that can take advantage of the union of these types of data. Put briefly, we are in a situation where we already have available a rich, mixed-media set of historical medical data but where we lack the machine-learning methods to take advantage of these data. Research on new algorithms that use such mixed-media data sets in medicine and other domains could lead to substantially more accurate methods for learning from available historical data and to substantially more accurate decision-making aids for physicians and other decision makers.

Learning using private data plus internet-accessible data: To illustrate this topic, consider a university interested in using historical data to identify current students who are at high risk of dropping out because of academic or other difficulties. The university might collect data on students over several years and apply a rule-learning method similar to that illustrated in figure 1. Such first-generation learning methods might work well, provided there is a sufficiently large set of training data and a data set that captures the features that are actually relevant to predicting student attrition. Now imagine a learning method that could augment the data collected for this study by also using the rich set information available over the World Wide Web. For example, the web might contain much more complete information about the particular academic courses taken by the student, the extracurricular clubs they joined, their living quarters, and so on. The point here is that the internet can provide a vast source of data that goes far beyond what will be available in any given survey. If this internet information can be used effectively, it can provide important additional features that can improve the accuracy of learned rules. Of course, much of this web-based information appears in text form, and it contains a very large number of irrelevant facts in addition to the small number of relevant facts. Research on new methods that utilize the web and other public online sources (for example, news feeds, electronic discussion groups) to augment available private data sets holds the potential to significantly increase the accuracy of learning in many applications.

Learning to Perceive: Face Recognition

A second application niche for machine learning lies in constructing computer programs that are simply too difficult to program by hand. Many sensor-interpretation problems fall into this category. For example, today's top speech-recognition systems all use machine-learning methods to some degree to improve their accuracy. A second example is image-classification problems, such as the face-recognition task illustrated in figure 3. The face images shown here have been used to train neural networks to classify new face images according to the identity of the person, his/her gender, and the direction in which he/she is looking. These images are part of a larger collection containing 624 images of 20 different people (selected students from Carnegie Mellon University's machine-learning class). For each person, approximately 32 different grey-scale images were collected, varying the person's expression, the direction in which he/she is looking, and his/her appearance with or without sunglasses. Given these data, students in the machine-learning class were asked to train a computer program to correctly recognize the face in a new image. Students succeeded in training a neural network to achieve this task with an accuracy of 90 percent at recognizing these faces compared to the 5-percent accuracy that would be obtained by randomly guessing which of the 20 people is in the image. In contrast, it is extremely difficult to manually construct a face-recognition program of comparable accuracy.

To illustrate the use of neural networks for image classification, consider the slightly simpler task of classifying these same images according to the direction in which the person is looking (for example, to his/her right or left or up or straight ahead). This task can also be accomplished with approximately 90-percent accuracy by training the neural network shown at the top left of figure 3. The input to the neural network encode the image, and the output classify the input image according to the direction in which the person is looking. More specifically, the image is represented by a 30 × 32 coarse-grained grid of pixel values, and each of these 960 pixel values is used as a distinct input to the network. The first layer of the network shown in the figure contains 3 units, each of which has the same set of 960 input. The activation of each such unit is computed by calculating a linear-weighted sum of the 960 input, then passing the result through a sigmoid-squashing function that maps this linear sum into the interval [0,1]. The activations of these three units are then input to the final layer of four output units. As shown in the figure, each output unit represents one of the four possible face directions, with the highest-valued output taken as the network prediction. This network is trained using the back-propagation algorithm (Rummelhart and McClelland 1986) to adjust the weights of each network unit so that the difference between the global network output and the known correct value for each training example is minimized.

One intriguing property of such multilayer neural networks is that during training, the internal hidden units develop an internal representation of the input image that captures features relevant to the problem at hand. In a sense, the learning algorithm discovers a new representation of the input image that captures the most important features. We can examine this discovered representation in the current example by looking at the 960 input weights to each of the 3 internal units in the network of figure 3. Suppose we display the 960 weight values (one for weighting each input pixel) in a grid, with each weight displayed in the position of the corresponding pixel in the image grid. This visualization is exactly the visualization of network weights shown to the right of the network in the figure. Here, large positive weights are displayed in bright white, large negative weights in dark black, and near-zero weights in grey. The topmost four blocks depict the weights for each of the four output units, and the bottom three large grids depict the

A second application niche for machine learning lies in constructing computer programs that are simply too difficult to program by hand.

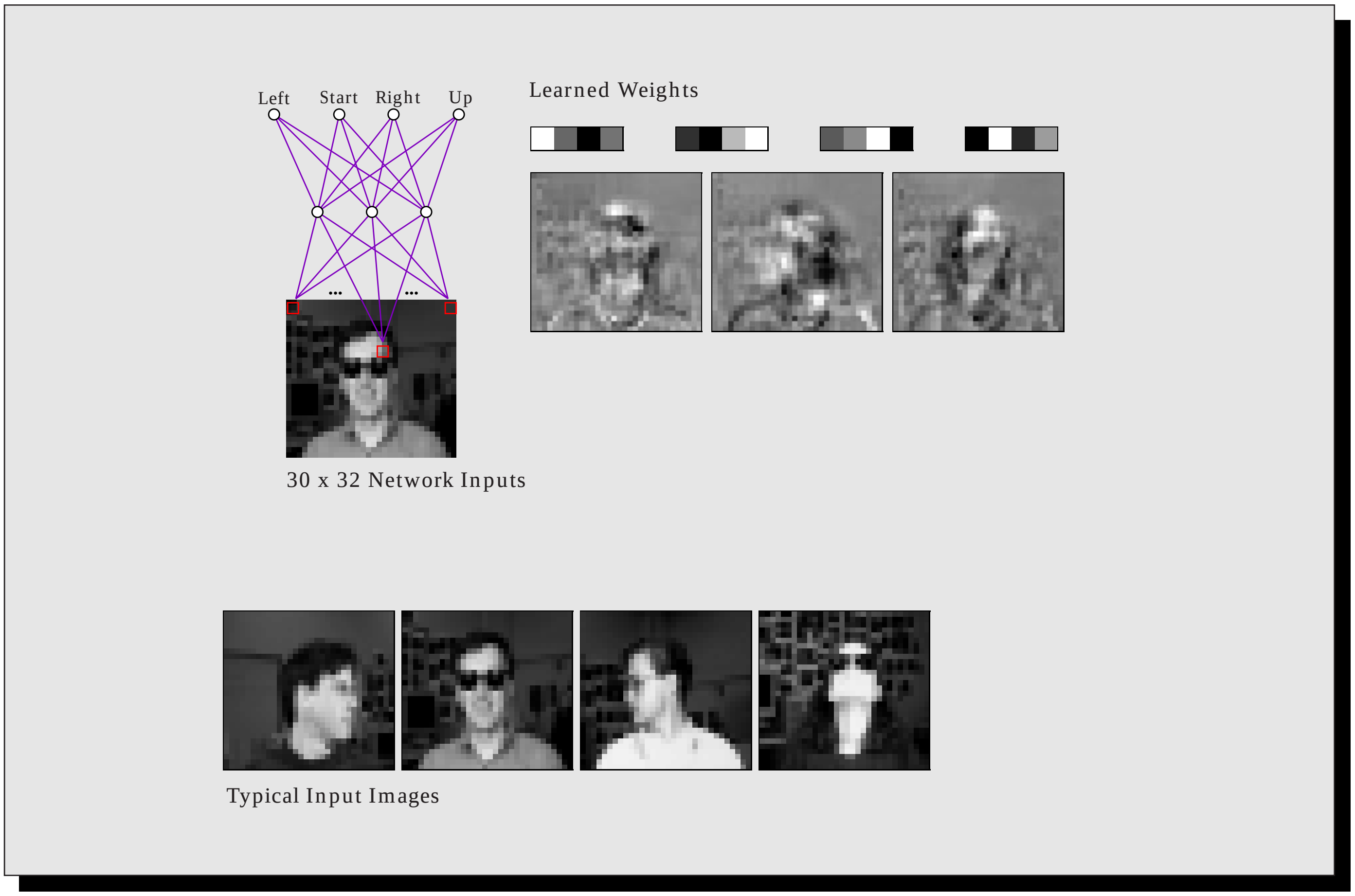


Figure 3. Face Recognition.

A neural network (top left) is trained to recognize whether the person is looking to his/her left or right or straight ahead or up. Each of the 30 × 32 pixels in the coarse-grained image is a distinct input to the network. After training on 260 such images, the network achieves an accuracy of 90 percent classifying future images. Similar accuracy is obtained when training to recognize the individual person. The diagrams on the top right depict the weight values of the learned network. These data and the learning code can be obtained at www.cs.cmu.edu/~tom/faces.html.

weights for the three internal, or hidden, units.

To understand these weight diagrams, consider first the four blocks at the top right, which describe the weights for the four output units of the network. Let us look at the third block, which encodes the weights for the third output unit, which represents that the person is looking to his/her right. This block shows four weight values. The first (leftmost) of these is the weight that determines the unit's threshold, and the remaining three are the weights connecting the three hidden units to this output unit. Note this "right" output unit has a large positive (bright-white) weight connecting to the middle hidden unit and a large negative (dark-black) weight connecting to the third hidden unit. Therefore, this unit will predict that the person is facing to his/her right if the second hidden unit has a high activation, and the third hidden unit does not.

Now consider the second hidden unit, whose activation will cause the correct output unit to activate. The weights for this unit are depicted in the middle diagram of the large weight diagrams directly below the output unit diagrams. Which pixels in the input image is this middle unit most sensitive to? Note first that the largest weight values (bright white and dark black) for this unit are associated with pixels that cover the face or body of the person, and the smallest (light grey) weights correspond to the image background. Thus, the face classification will depend primarily on this section of the image. Second, notice the cluster of bright (positive) weight values near the left side of the "face" and the cluster of dark (negative) weight values near the right side. Thus, this particular unit will produce a large weighted sum when it is given an input image with bright pixels overlapping

True Rating	**Predicted Rating**						
	1	2	3	4	5	Skip	Total
1:	0	1	0	0	0	1	2
2:	1	15	6	4	0	15	41
3:	0	6	31	20	0	15	72
4:	0	6	8	42	0	20	76
5:	0	0	0	4	0	1	5
skip:	0	8	4	5	1	141	159

Table 1. Results of NEWS WEEDER *for One User.*

After observing the user read approximately 1300 news articles, the system predicted ratings for approximately 600 new articles. The previous confusion matrix shows the relationship between NEWS WEEDER's predictions and the actual user ratings. For example, the 20 in the table indicates there were 20 articles for which NEWS WEEDER predicted a rating of 4 but that the user rated a 3. Note the large numbers along the diagonal indicate correct predictions.

these positive weights and dark pixels overlapping these negative weights. One example of such an image is the third face image shown at the bottom of the figure, in which the person is looking to his/her right. This unit will therefore have a high output when given the image of the person looking to his/her right, whereas it will have a low value when given an image of a person looking to his/her left (for example, the first image shown in the figure).

Neural network–learning algorithms such as the back-propagation algorithm given here are used in many sensor-interpretation problems. If you would like to experiment for yourself with this face-recognition data and learning algorithm, the data, code, and documentation are available at www.cs.cmu.edu/~tom/faces.html.

What is the state of the art in applying machine learning to such perception problems? In many perception problems, learning methods have been applied successfully and outperform nonlearning approaches (for example, in many image-classification tasks and other signal-interpretation problems). Still, there remain substantial limits to the current generation of learning methods. For example, the straightforward application of back propagation illustrated in figure 3 depends on the face occurring reliably in the same portion of the image. This network cannot easily learn to recognize faces independent of translation, rotation, or scaling of the face within the image. For an example of how such neural network face-recognition systems can be extended to accommodate translation and scaling, see Rowley, Baluja, and Kanade (1996).

There is a great deal of active research on learning in perception and on neural network learning in general. One active area of new research involves developing computational models that more closely fit what is known about biological perceptual systems. Another active area involves attempts to unify neural network–learning methods with Bayesian learning methods (for example, learning of Bayesian belief networks). One of my favorite areas for new thesis topics is long-term learning of new representations. Because of the ability of neural networks to learn hidden-layer representations, as illustrated in figure 3, they form one good starting point for new research on long-term learning of representations.

A Self-Customizing News Reader

A third niche for machine learning is computer software that automatically customizes to its users after it has been fielded. One prototypical example is the NEWS WEEDER system (Lang 1995), an interface to electronic news groups that learns the reading interests of its users. Here, we briefly summarize NEWS WEEDER and its learning method.

The purpose of NEWS WEEDER is to learn to filter the flood of online news articles on behalf of its user. Each time the user reads an article using NEWS WEEDER, he/she can assign it a rating from 1 (very interesting) to 5 (uninteresting) to indicate interest in the article. NEWS WEEDER uses these training examples to learn a general model of its user's interests. It can then use this learned interest model to automatically examine new articles, or articles from other news groups, to assemble a "top 20" list for the user.

Given that there are thousands of news groups available online and that most users read only a few, the potential benefit is that the agent can perform the task of sorting through the vast bulk of uninteresting articles to bring the few interesting ones to the attention of its user.

The key machine-learning question underlying NEWS WEEDER is, How can we devise algorithms capable of learning a general model of user interests from a collection of specific text documents? NEWS WEEDER takes an approach that is based both on earlier work in information retrieval and on Bayesian learning methods. First, each text article is re-represented by a very long feature vector, with one feature for each possible word that can occur in the text. Given that there are approximately 50,000 words in standard English, this feature vector contains approximately 50,000 features. The value for each feature (for example, the feature corresponding to the word *windsurf*) is the number of times the word occurs in the document. This feature-vector representation, based on earlier work in information retrieval, is sometimes called a *bag of words representation* for the document because it omits information about word sequence, retaining only information about word frequency.

Once the document is represented by such a feature vector, NEWS WEEDER applies a variant of a naive Bayes classification learning method to learn to classify documents according to their interestingness rating. In the naive Bayes approach, the system assigns a rating to each new article based on several probability terms that it estimates from the training data, as described in the following paragraph.

During training, the naive Bayes classifier determines several probabilities that are used to classify new articles. First, it determines the prior probability $P(C_i)$ for each possible rating C_i by calculating the frequency with which this rating occurs over the training data. For example, if the user assigned a top rating of 1 to just 9 of the 100 articles he/she read, then $P(Rating = 1) = 0.9$. Second, it estimates the conditional probability $P(w_j \mid C_i)$ of encountering each possible word w_j given that the document has a true rating of C_i. For example, if the word *windsurf* occurs 11 times among the 10,000 words appearing in articles rated 1 by the user, then $P(windsurf \mid Rating = 1) = .0011$. A new text document d is then assigned a new rating by choosing the rating C_i that maximizes

$$P(C_i)\prod_{w_j \in d} P(w_j \mid C_i)$$

where $\Pi_{w_j \in d}\, P(w_j \mid C_i)$ represents the product of the various $P(w_j \mid C_i)$ over all words w_j found in the document d. As we might expect, the higher the conditional probabilities $P(w_j \mid C_i)$ for the words w_j that are actually encountered in d, the greater the chance that C_i is the correct rating for d.

In fact, NEWS WEEDER's learning method uses a variant of this algorithm, described in Lang (1995). One difference is that the document length is taken into account in estimating the probability of word occurrence. A second is that the system combines the strength of its predictions for each rating level, using a linear combination of these predictions to produce its final rating.

How well does NEWS WEEDER learn the user's interests? In one set of experiments, the system was trained for one user using a set of approximately 1900 articles. For each article the user read, a rating of 1 to 5 was assigned. Articles that the user chose not to read were labeled *skip*. A subset of approximately 1300 of these articles was used to train the system, and its performance was then evaluated over the remaining articles. NEWS WEEDER's predicted ratings are compared to the actual user ratings in table 1. Note that numbers along the diagonal of this confusion matrix indicate cases where the predicted and true ratings were identical, whereas off-diagonal entries indicate differences between these two. As is apparent from table 1, NEWS WEEDER does a fairly good job of predicting user interest in subsequent documents. Note that in many cases, its errors represent small differences between the predicted and actual ratings (for example, many of its errors when predicting a rating of 4 were articles that were actually rated 3 by the user).

NEWS WEEDER illustrates the potential role of machine learning in automatically customizing software to users' needs. It also illustrates how machine learning can be applied to the growing volume of text data found on electronic news feeds, the World Wide Web, and other online sources. Because the initial experiments with NEWS WEEDER were promising, a second-generation system called WISE WIRE has now been developed at WiseWire Corporation. This system is available free of charge at www.wisewire.com.

The current state of the art for learning to classify text documents is fairly well represented by the naive Bayes text classifier described here. An implementation of this algorithm for text classification can be obtained at www.cs.cmu.edu/~tom/ml-examples.html. Machine learning over text documents is an active area of current research. One interesting research topic involves going beyond the bag-of-words approach, representing the text by more linguistic features such as noun phrases and partial interpretations of the meaning of the text. In addition to classifying text documents, other important text-learning problems include learning to extract information from text (for example, learning to extract the title, speaker, room, and subject, given online seminar announcements). Given the rapidly growing importance of the World Wide Web and other online text such as news feeds, it seems likely that machine learning over text will become a major research thrust over the coming years.

Where Next?

As the previous three examples illustrate, machine learning is already beginning to have a significant impact in several niches within computer science. My personal belief is that the impact of machine learning is certain to grow over the coming years, as more and more data come online, we develop more effective algorithms and underlying theories for machine learning, the futility of hand-coding increasingly complex systems becomes apparent, and human organizations themselves learn the value of capturing and using historical data.

Put in a long-term historical perspective, computer science is a young field still in its first 50 years. Perhaps we will look back after the first hundred years to see that the evolution of computer science followed a predictable course: During its first few decades, computer science focused on fairly simple algorithms and computer programs and developed a basic understanding of data structures and computation. During the next several decades, it struggled with the problem of how to construct increasingly complex software systems but continued to follow its early approach of manually hand crafting each line of code. During the second half of its first century (hopefully!), a shift is seen to take place in which software developers eventually abandon the goal of manually hand crafting increasingly complex software systems. Instead, they shift to a software development methodology of manually designing the global structure of the system but using machine-learning methods to automatically train individual software components that cannot be found in a software library and to optimize global performance. Of course, we will have to wait to find out how the second 50 years pans out, but at least for the three niches discussed here, it is already the case that machine learning competes favorably with manual hand-crafted code.

As for us researchers, many exciting opportunities are available right now regardless of how the next 50 years pan out. In addition to some of the specific research directions noted earlier, I believe there are several areas where significant breakthroughs are possible. Below are some of my speculations on which research areas might produce dramatic improvements in the state of the art of machine learning.

Incorporation of prior knowledge with training data: One striking feature of the learning methods described earlier is that they are all *tabula rasa methods*; that is, they assume the learner has no initial knowledge except for the representation of hypotheses and that it must learn solely from large quantities of training data. It is well understood that prior knowledge can reduce sample complexity and improve learning accuracy. In fact, machine-learning algorithms such as explanation-based learning (DeJong 1997) and, more recently, Bayesian belief networks provide mechanisms for combining prior knowledge with training data to learn more effectively. Nevertheless, it remains a fact that almost all the successful uses of machine learning to date rely only on tabula rasa approaches. More research is needed to develop effective methods of combining prior knowledge with data to improve the effectiveness of learning.

Lifelong learning: Most current learning systems address just one task. However, humans learn many tasks over a long lifetime and seem to use the accumulating knowledge to guide subsequent learning. One interesting direction in current research is to develop long-life agents that accumulate knowledge and reuse it to guide learning in a similar fashion. These agents might learn new representations, as well as facts, so that they can more effectively represent and reason about their environment as they age. Several researchers are already examining this topic in the context of robot learning and autonomous software agents.

Machine learning embedded in programming languages: If one assumes that machine learning will play a useful and routine role within certain types of computer software, then it is interesting to ask how future software developers will incorporate learning into their software. Perhaps it is time for the development of new programming languages or programming environments that allow developers to incorporate machine learning into their software as easily as current environments allow incorporating abstract data types and other standard software constructs.

Machine learning for natural language: One fact that seems certain to influence machine learning is the fact that the majority of the world's online data is now text. Given approximately 200,000,000 pages on the

… the impact of machine learning is certain to grow over the coming years, as more and more data come online, we develop more effective algorithms and underlying theories for machine learning, the futility of hand-coding increasingly complex systems becomes apparent, and human organizations themselves learn the value of capturing and using historical data.

World Wide Web, and approximately 1,000,000,000 hyperlinks, it might be time to consider machine learning for natural language problems. Consider that each of these billion hyperlinks is a kind of semisupervised training example that indicates the meaning of the words underlined by the hyperlink. For example, the meaning of a hyperlink such as *my favorite vacation spot* is indicated by the web page to which it points. Given that there are only 50,000 words in everyday English, the 1,000,000,000 hyperlinks produce, on average, 20,000 training examples of each word. If we could invent learning algorithms that use this kind of training data, we might be able to make significant new progress on natural language processing.

Acknowledgments

The research described here is the result of the efforts of a number of people and organizations. Schenley Park Research, Inc., provided the medical data-mining example. Jeff Shufelt, Matt Glickman, and Joseph O'Sullivan helped create the code and collect the data for the face-recognition example. This work has been supported in part by the Defense Advanced Research Projects Agency under research grant F33615-93-1-1330.

Parts of this article are excerpted from the textbook *Machine Learning* (Mitchell 1997).

References

Aha, D.; Kibler D.; and Albert, M. 1991. Instance-Based Learning Algorithms. In *Machine Learning 6*, 37–66. New York: Kluwer Academic.

Barto, A.; Bradtke, S.; and Singh, S. 1995. Learning to Act Using Real-Time Dynamic Programming. *Artificial Intelligence* (Special Issue on Computational Research on Interaction and Agency) 72(1): 81–138.

Bishop, C. M. 1996. *Neural Networks for Pattern Recognition.* Oxford, U.K.: Oxford University Press.

Caruana, R.; Baluja, S.; and Mitchell, T. 1995. Using the Future to Sort Out the Present: RANKPROP and Multitask Learning for Medical Risk Analysis. In *Neural Information Processing* 7, eds. G. Tesauro, D. S. Touretzky, and T. K. Leen. Cambridge, Mass.: MIT Press.

Chauvin, Y., and Rumelhart, D. 1995. *Backpropagation: Theory, Architectures, and Applications.* Hillsdale, N.J.: Lawrence Erlbaum.

Clark, P., and Niblett, R. 1989. The CN2 Induction Algorithm. In *Machine Learning 3*, 261–284. New York: Kluwer Academic.

DeJong, G. 1997. Explanation-Based Learning. In *The Computer Science and Engineering Handbook,* ed. A. Tucker, 499–520. Boca Raton, Fla.: CRC Press.

Forrest, S. 1993. Genetic Algorithms: Principles of Natural Selection Applied to Computation. *Science* 261:872–878.

Hecht-Nielsen, R. 1990. *Neurocomputing.* Reading, Mass.: Addison Wesley.

Joachims, T.; Freitag, D.; and Mitchell, T. 1997. WEBWATCHER: A Tour Guide for the World Wide Web. In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence. Forthcoming.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement Learning: A Survey. *Journal of AI Research* 4:237–285. Available at www.cs.washington.edu/research/jair/home.html.

Kearns, M. J., and Vazirani, U. V. 1994. *An Introduction to Computational Learning Theory.* Cambridge, Mass.: MIT Press.

Lang, K. 1995. NEWS WEEDER: Learning to Filter NETNEWS. In *Proceedings of the Twelfth International Conference on Machine Learning,* eds. A. Prieditis and S. Russell, 331–339. San Francisco, Calif.: Morgan Kaufmann.

Lavraĉ, N., and Dẑeroski, S. 1994. *Inductive Logic Programming: Techniques and Applications.* Ellis Horwood.

Mitchell, T. M. 1997. *Machine Learning.* New York: McGraw Hill.

Mitchell, M. 1996. *An Introduction to Genetic Algorithms.* Cambridge, Mass.: MIT Press.

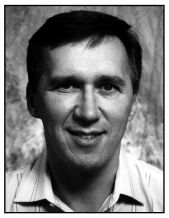
Mitchell, T. M.; Caruana, R.; Freitag, D.; McDermott, J.; and Zabowski, D. 1994. Experience with a Learning Personal Assistant. *Communications of the ACM* 37(70): 81–91.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning.* San Francisco, Calif.: Morgan Kaufmann.

Quinlan, J. R. 1990. Learning Logical Definitions from Relations. In *Machine Learning 5*, 239–266. New York: Kluwer Academic.

Rowley, H.; Baluja, S.; and Kanade, T. 1996. Human Face Detection in Visual Scenes. In *Advances in Neural Information Processing Systems* 8, eds. D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo. Cambridge, Mass.: MIT Press.

Rumelhart, D. E., and McClelland, J. L. 1986. *Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Volumes 1 and 2.* Cambridge, Mass.: MIT Press.



Tom Mitchell is currently professor of computer science and robotics at Carnegie Mellon University (CMU) and director of the CMU Center for Automated Learning and Discovery. He recently completed a comprehensive textbook entitled *Machine Learning* (McGraw Hill, 1997). Mitchell's current research involves applying machine learning to the vast store of undisciplined information found on the World Wide Web and other internet sources. His e-mail address is tom.mitchell@cmu.edu.