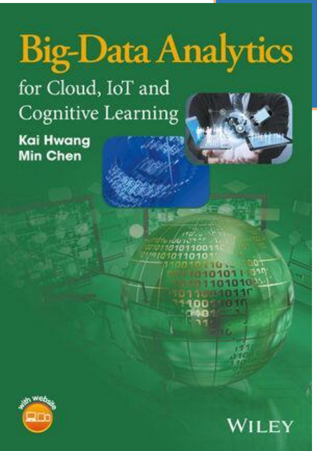
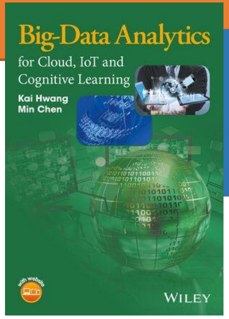
Big Data Analytics for Cloud, IoT and Cognitive Computing

Part 3 Big Data Analytics for Health-Care and Cognitive Learning

Chap. 8 Deep Reinforcement Learning and Social Media Analytics





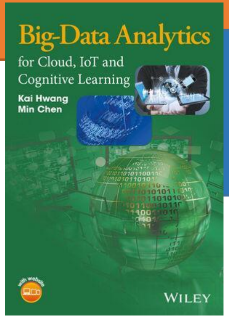
Deep Learning Systems and Social Media Industry

The software libraries or platforms were developed by industry and academia for machine learning (ML) and deep learning (DL) applications.

As studied in earlier chapters, deep learning is part of a broader family of machine learning methods. The difference lies in the learning representations of data.

For example:

- the inspection of an X-ray image is represented in many ways such as vectors, matrices or tensors. Some of the representations are inspired by advances in neuroscience.



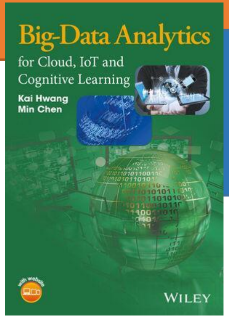
Deep Learning Systems and Software Support

We have learned machine learning and deep learning algorithms. With huge amounts of data to train the model to perfection, the clouds have sufficient resources to achieve the learning processes in real time. Deep learning is certainly an extended field of machine learning.

Software, Creator, Language(s)/ Interface, License, and Web Site	Platform(s), Software Tools	DL Models Supported	A Brief Description
Caffe, Berkeley Vison and Learning Center, C++, Python, MATLAB, BSD2 , http://caffe.berkeleyvision.org/	AWS, OSX, Windows, OpenCL, CUDA,	RNN, CNN	A DL framework adopts a pure C++/ CUDA architecture for easy switch between CPU and GPU.
CNTK, Microsoft, C++, Python, .NET, Free, http://github.com/ Microsoft/CNTK	Windows, Linux, OpenMP, CUDA,	RNN, CNN	A free DL software for cross platform applications.
TensorFlow, Google Brain Team, C/C++, Python, Apache 2.0, https://www.tensorflow.org/	OpenCL on roadmap, CUDA,	RNN, CNN, RBM, DBN	Based on DistBelief, allowing tensor to flowthrough ANN graph from one end to another.
Theano, U.of Montreal, SD, Python http://deeplearning.net/ software/theano	Cross platform, Open MP, CUDA	RNN, CNN, RBM, DBN	A framework using Torch for a modular ANN library supporting joint GPU and CPU operations.
Torch, Ronan Collobert, C, Lua, BSD, http://torch.ch/	Linux, Android, OSX, iOS, OpenCL	RNN, CNN, RBM, DBN	Built with iTorch and fbcunn, to improve ANN performance in computer vision and natural language processing.

Table 8.1: Comparison of Five Open Source Software Libraries for Deep Learning Application

Five deep learning software libraries are compared in Table 8.1 in terms of modeling capability, interfaces, performance, cross platform, and reported performance.



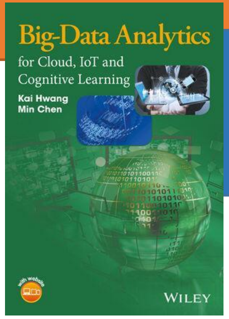
Deep Learning Systems and Software Support

We have learned machine learning and deep learning algorithms. With huge amounts of data to train the model to perfection, the clouds have sufficient resources to achieve the learning processes in real time. Deep learning is certainly an extended field of machine learning.

Software, Creator, Language(s)/ Interface, License, and Web Site	Platform(s), Software Tools	DL Models Supported	A Brief Description
Caffe, Berkeley Vison and Learning Center, C++, Python, MATLAB, BSD2 , http://caffe.berkeleyvision.org/	AWS, OSX, Windows, OpenCL, CUDA,	RNN, CNN	A DL framework adopts a pure C++/ CUDA architecture for easy switch between CPU and GPU.
CNTK, Microsoft, C++, Python, .NET, Free, http://github.com/ Microsoft/CNTK	Windows, Linux, OpenMP, CUDA,	RNN, CNN	A free DL software for cross platform applications.
TensorFlow, Google Brain Team, C/C++, Python, Apache 2.0, https://www.tensorflow.org/	OpenCL on roadmap, CUDA,	RNN, CNN, RBM, DBN	Based on DistBelief, allowing tensor to flowthrough ANN graph from one end to another.
Theano, U.of Montreal, SD, Python http://deeplearning.net/ software/theano	Cross platform, Open MP, CUDA	RNN, CNN, RBM, DBN	A framework using Torch for a modular ANN library supporting joint GPU and CPU operations.
Torch, Ronan Collobert, C, Lua, BSD, http://torch.ch/	Linux, Android, OSX, iOS, OpenCL	RNN, CNN, RBM, DBN	Built with iTorch and fbcunn, to improve ANN performance in computer vision and natural language processing.

Table 8.1: Comparison of Five Open Source Software Libraries for Deep Learning Application

Caffe is the most popular toolkit within the computer vision community. In general, its support for recurrent networks and language modeling is limited due to its legacy architecture. Caffe has pycaffe interface but that's a mere secondary alternative to the command line interface.



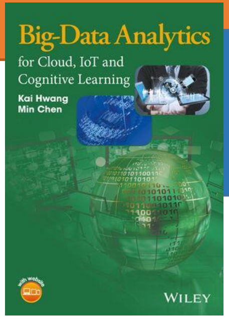
Deep Learning Systems and Software Support

We have learned machine learning and deep learning algorithms. With huge amounts of data to train the model to perfection, the clouds have sufficient resources to achieve the learning processes in real time. Deep learning is certainly an extended field of machine learning.

Software, Creator, Language(s)/ Interface, License, and Web Site	Platform(s), Software Tools	DL Models Supported	A Brief Description
Caffe, Berkeley Vison and Learning Center, C++, Python, MATLAB, BSD2 , http://caffe.berkeleyvision.org/	AWS, OSX, Windows, OpenCL, CUDA,	RNN, CNN	A DL framework adopts a pure C++/ CUDA architecture for easy switch between CPU and GPU.
CNTK, Microsoft, C++, Python, .NET, Free, http://github.com/ Microsoft/CNTK	Windows, Linux, OpenMP, CUDA,	RNN, CNN	A free DL software for cross platform applications.
TensorFlow, Google Brain Team, C/C++, Python, Apache 2.0, https://www.tensorflow.org/	OpenCL on roadmap, CUDA,	RNN, CNN, RBM, DBN	Based on DistBelief, allowing tensor to flowthrough ANN graph from one end to another.
Theano, U.of Montreal, SD, Python http://deeplearning.net/ software/theano	Cross platform, Open MP, CUDA	RNN, CNN, RBM, DBN	A framework using Torch for a modular ANN library supporting joint GPU and CPU operations.
Torch, Ronan Collobert, C, Lua, BSD, http://torch.ch/	Linux, Android, OSX, iOS, OpenCL	RNN, CNN, RBM, DBN	Built with iTorch and fbcunn, to improve ANN performance in computer vision and natural language processing.

Table 8.1: Comparison of Five Open Source Software Libraries for Deep Learning Application

Like TensorFlow and Theano, CNTK is specified as a symbolic graph of vector operations, such as matrix add/multiply or convolution. A layer in the neural network is just a composition of those operations.



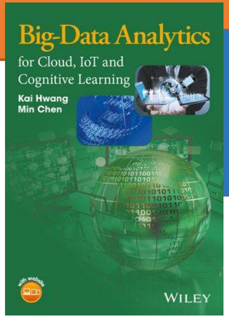
Deep Learning Systems and Software Support

We have learned machine learning and deep learning algorithms. With huge amounts of data to train the model to perfection, the clouds have sufficient resources to achieve the learning processes in real time. Deep learning is certainly an extended field of machine learning.

Software, Creator, Language(s)/ Interface, License, and Web Site	Platform(s), Software Tools	DL Models Supported	A Brief Description
Caffe, Berkeley Vison and Learning Center, C++, Python, MATLAB, BSD2 , http://caffe.berkeleyvision.org/	AWS, OSX, Windows, OpenCL, CUDA,	RNN, CNN	A DL framework adopts a pure C++/ CUDA architecture for easy switch between CPU and GPU.
CNTK, Microsoft, C++, Python, .NET, Free, http://github.com/ Microsoft/CNTK	Windows, Linux, OpenMP, CUDA,	RNN, CNN	A free DL software for cross platform applications.
TensorFlow, Google Brain Team, C/C++, Python, Apache 2.0, https://www.tensorflow.org/	OpenCL on roadmap, CUDA,	RNN, CNN, RBM, DBN	Based on DistBelief, allowing tensor to flowthrough ANN graph from one end to another.
Theano, U.of Montreal, SD, Python http://deeplearning.net/ software/theano	Cross platform, Open MP, CUDA	RNN, CNN, RBM, DBN	A framework using Torch for a modular ANN library supporting joint GPU and CPU operations.
Torch, Ronan Collobert, C, Lua, BSD, http://torch.ch/	Linux, Android, OSX, iOS, OpenCL	RNN, CNN, RBM, DBN	Built with iTorch and fbcunn, to improve ANN performance in computer vision and natural language processing.

Table 8.1: Comparison of Five Open Source Software Libraries for Deep Learning Application

TensorFlow supports state-of-the-art models: RNN, CNN, RBM, and DBN; so do Theano and Torch.



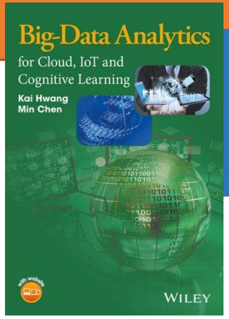
Deep Learning Systems and Software Support

We have learned machine learning and deep learning algorithms. With huge amounts of data to train the model to perfection, the clouds have sufficient resources to achieve the learning processes in real time. Deep learning is certainly an extended field of machine learning.

Software, Creator, Language(s)/ Interface, License, and Web Site	Platform(s), Software Tools	DL Models Supported	A Brief Description
Caffe, Berkeley Vison and Learning Center, C++, Python, MATLAB, BSD2 , http://caffe.berkeleyvision.org/	AWS, OSX, Windows, OpenCL, CUDA,	RNN, CNN	A DL framework adopts a pure C++/ CUDA architecture for easy switch between CPU and GPU.
CNTK, Microsoft, C++, Python, .NET, Free, http://github.com/ Microsoft/CNTK	Windows, Linux, OpenMP, CUDA,	RNN, CNN	A free DL software for cross platform applications.
TensorFlow, Google Brain Team, C/C++, Python, Apache 2.0, https://www.tensorflow.org/	OpenCL on roadmap, CUDA,	RNN, CNN, RBM, DBN	Based on DistBelief, allowing tensor to flowthrough ANN graph from one end to another.
Theano, U.of Montreal, SD, Python http://deeplearning.net/ software/theano	Cross platform, Open MP, CUDA	RNN, CNN, RBM, DBN	A framework using Torch for a modular ANN library supporting joint GPU and CPU operations.
Torch, Ronan Collobert, C, Lua, BSD, http://torch.ch/	Linux, Android, OSX, iOS, OpenCL	RNN, CNN, RBM, DBN	Built with iTorch and fbcunn, to improve ANN performance in computer vision and natural language processing.

Table 8.1: Comparison of Five Open Source Software Libraries for Deep Learning Application

- Theano has implementation for most state-of-the-art neural networks. This software pioneered the trend of using symbolic graph for programming a network.
- Torch supports convolutional neural networks nicely. The native interface for temporal convolution in Torch makes it more intuitive to use. Torch runs on LuaJIT, which is very fast.

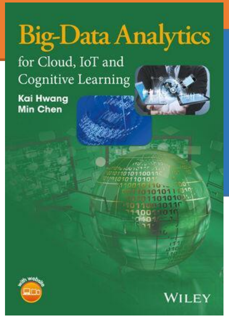


Deep Learning Systems and Software Support

Certainly, deep learning with ANNs has been proven some impressive successes in recent years. A lot of interesting ML and DL products and systems are developed by industry and academia:

- Microsoft Office 365 offers the cloud-based Outlook Web Access (OWA) to manage the eMails of tens of thousands of organizations.
- ICloud offers free email services to lock up their iPhone ot iPad users. Other interesting products include Apple's Siri and fingerprint ID, Google's WatsApp and Scholars, SalesForce CRM services, Tencent's WeChat, and Microsof Skype, etc.
- On March 15, 2016, Google introduced the *Google Analytics 360 Suite*.

Besides speech/language processing, machine perception, automated image understanding, vision analysis, and AR/VR products are very hot products in the IT and entertainment industry.



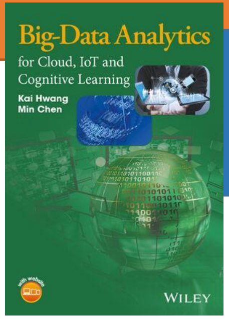
Reinforcement Learning Principles

The RL is indeed a subclass of unsupervised ML, because the correct input/output pairs are never presented. Reinforcement learning is considered as a general-purpose framework of artificial intelligence.

Formally, an RL model is built with the following 5 constituting parts:

- A learning environment characterized by a set of states.
- A set of actions that can be taken by RL agents. Each action influences the agent's future states. The agent has the capacity to assess the long-term consequences of its actions.
- Rules of transitions between the RL states
- Rules that determine the immediate reward of a state transition
- Rules that specify what the agent can observe.

RL algorithms have been applied successfully in robot control, elevator scheduling, cognitive radio, and game playing like chess, checkers, go and Atari games, etc. In a nutshell, the idea is to select actions to maximize the future rewards.



Reinforcement Learning Methods

- RL algorithms encourage the use of samples to optimize performance and the use of function approximation to deal with large environments. These two features make RL especially effective in handling three machine learning environments:

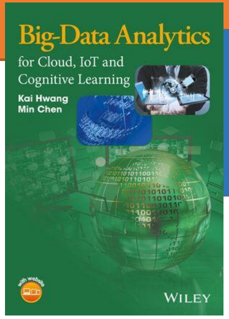
  (1) A known model environment short of an analytic solution

  (2) A simulation-based optimization environment

  (3) An environment enabling information collecting by interacting with it.

- Fundamental assumptions on the reinforcement learning environments include the following:

  - All events are episodic as a sequence of episodes. An episode ends when some terminal state is reached.
  - No matter what course of actions the agent may take, termination is inevitable.
  - The expectation of the total reward is well-defined for any policy and any initial distribution over the states.

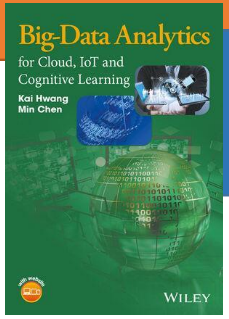


Reinforcement Learning Methods

An intelligent agent must be able to work out an RL algorithm to find a policy with maximum expected gain. The algorithm needs to search for optimal policy to achieve maximal rewards. Often, we apply deterministic stationary policies, which select the actions deterministically, based only on the current or last state visited. There are a number of approaches in designing reinforcement learning algorithms:

- A brute-force method is to to choose the policy with the largest expected return. The major difficulty with this approach is the policy choices could be very large or even infinite. Value function approaches attempt to find a policy that maximizes the return by maintaining a set of estimates of expected returns for some policy.

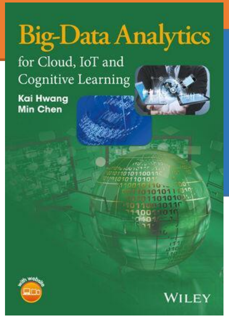
- Other RL schemes include the time-difference method which allows the policy to change before the award values are settled. A direct policy search method finds a good policy by searching directly from a policy space. Both gradient-based and gradient-free methods belong to this class.



Social-Media Industry and Global Impact

The social-media industry is moving away from flat media such as newspapers, magazines, and television shows. Instead, e-books, mobile payment, Uber cars, on-line shopping, and social networking are gradually becoming the main stream. The trick is to catch or target users at optimal times in ideal locations. The ultimate aim is to convey a message or content that is in line with the consumers mindset. For example, e-newspapers and e-books are replacing hardcopy books and newspapers.

Targeting of consumers is closely tied to data-capture method much more than in the past. This is best seen by finding the correlation of IoT and big data. Various IoT sensing technologies have transformed the way media industry, business companies and even governments operate. This has affected the economic growth and competitiveness.

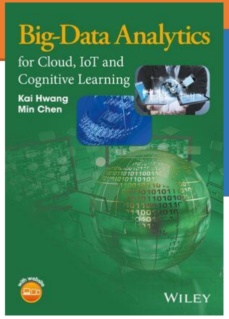


Social-Media Industry and Global Impact

The social media industry provides computer-mediated tools that allow people or companies to create, share, or exchange information, career interests, ideas, pictures and videos in all walks of life. According to Wikipedia Social media services are presented in the following 4 areas in our daily life activities:

- Social media services are part of Web 2.0 web service applications
- User-generated content is the lifeblood of the social media organism
- Users create service-specific profiles for social media organizations and websites.
- Social media facilitate the development of online social networks in societal and business activities.

Social media enables fundamental changes to communication between businesses, organizations, communities, and individuals. These changes demand the social media industry to operate from many sources to many receivers.



Social-Media Industry and Global Impact

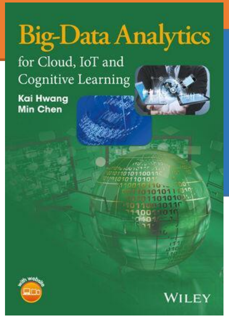
Example 9.1 Social-Media Applications Programming Interfaces (APIs)

Application programming interfaces (APIs) are the first software tools to access a computer, website or cloud platform. These APIs enable user or programmers to start using the system being programmed. Table 8.2 lists ten representative APIs for social-media big data applications. We characterize each API by its functionality, protocol, data format and security applied.

Table 8.2 Social-media application programming interfaces (APIs).

API Name	Functionality	Protocol applied	Data Format	Security
Facebook Graph API	Facebook social graph processing, community detection, and finding friends, etc. .	REST	JSON	OAuth
Google+ API	To provide access of Google+, a social media web site with link, status, and photo options.	REST	JSON	API key, OAuth
Social Mention API	Programmatic access to interact with Social Mention website, a RESTful API	HTTP	PHP	API key
Delicious API	Allow users to access, edit, and search for bookmarks	REST	JSON, RSS	OAuth, HTTP/Bas
MySpace API	To access various MySpace functions and integrate application into MySpace	Javascript	Unknown	OAuth
Meetup API	To use the Topics, Groups, and Events created by Meetup into their own applications	REST	JSON, XML KML, RSS	PAith, API key
FindMeOn API v.1,0	Programmtic access to the social media search and management functions of FibdMeOn.	HTTP	JSON	API key
Fliptop API	Person API to get social data based on eMail address, or utize Twitter/Facebook handles to elicit data return	REST	JSON, XML	API key
Cisco JTAPI	Cisco Java Telephony API allows Java appcations to interact with telephony resources	SOAP, HTTP	XML	SSL Support
YouTube Data API v3.0	Perform actions available on the YouTube website.	REST, HTTP	JSON	API key

All computer, cloud and social media providers have their own API tools. Among them, REST is known for the most popular protocol, JSON is the most used format, and API key for most security control. Those listed above are only some representative ones, as they are many more for various IT companies and social websites.



Text and Image Recognition using ANN and CNN

How to utilize artificial neural network (ANN) and convolution neural network (CNN) to achieve handwritten numeral recognition？ Handwritten numeral recognition is a classification problem.

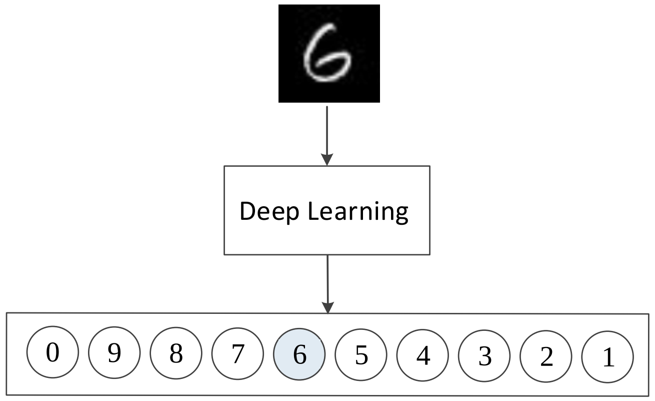
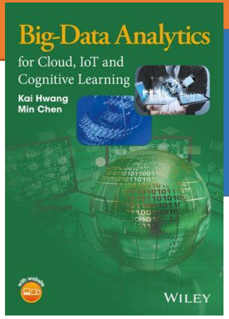


Figure 8.1 A deep learning system for recognizing handwritten numerals

The input is the handwritten numeral image and output represents the number expressed by the image. In order to allow readers to learn and practice easily, we use a classic handwritten numeral set Mnist as application data sets. Mnist includes 60,000 images of handwritten numerals, and each image is 28x28 pixels.



Numeral Recognition using TensorFlow To Program ANN

Example 8.2: Programming an ANN by TensorFlow

The following example shows how to use TensorFlow in programming an artificial neural network (ANN), called the Mnist classifier. We consider the construction of a 4-layer ANN, called the Mnist classifier. There are 4 steps to construct their ANN. We specify the procedure of each step, separately, using pseudo codes with comments, which are rather close to Python codes.

Step 1: Collect the data :

We use the Mnist data which is taken from Yann LeCun's website

Step 2: Construct the ANN model :

We select a 4-layer neural network to construct the classifier, which contains one input layer, 2 hidden layers and one output layer.

Step 3: Train the model.

By comparing the output of training data and its labels, the algorithm will adjust the parameters of the network.

Step 4: Test the network.

The algorithm will compare the output of the test data and its output of the test data and its corresponding label and calculate

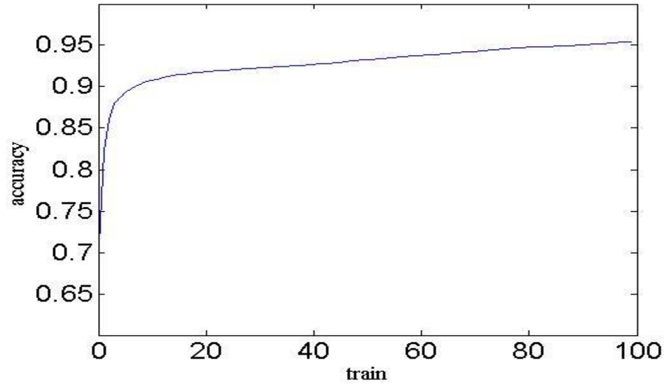
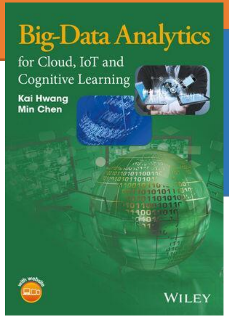


Figure 8.2: Results of TensorFlow based on programming an artificial neural network.



Numeral Recognition using TensorFlow To Program ANN

The intermediate layer extracts features and classifies with CNN, namely utilizing CNN to realize handwritten numeral recognition. With multiple convolution layers, pooling layers and a fully connected layer, the recognition result can be obtained in the output layer.

The structure of a 5-deep learning layers in a convolutional neural network (CNN) is shown in Fig. 8.3. There are two convolution layers in this structure. Each convolution layer is closely followed by a pooling layer. The last layer is a full connected layer and the output classification is conducted last.

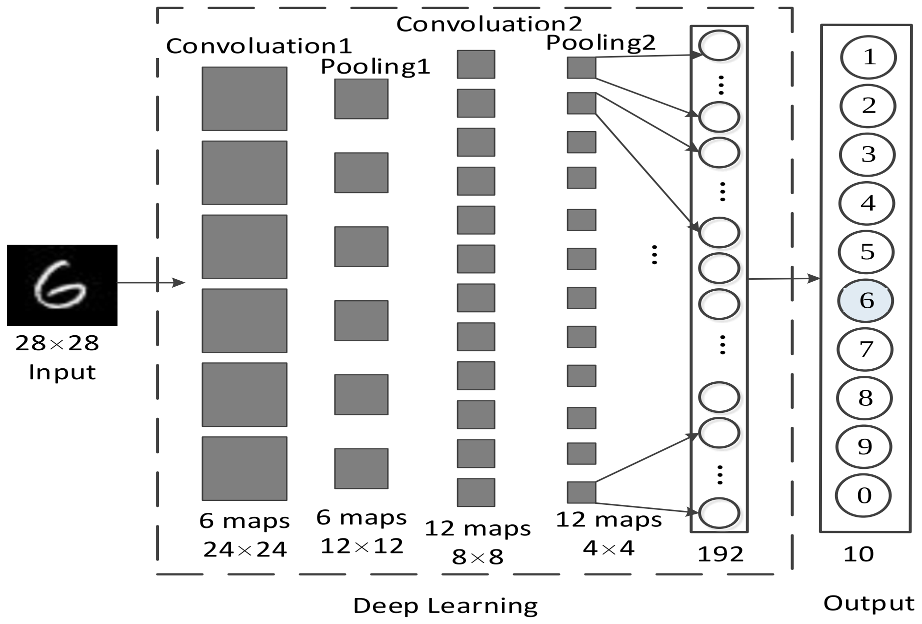
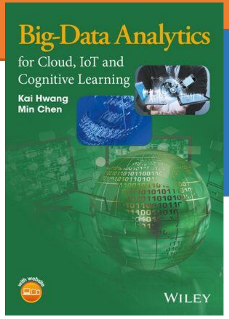


Figure 8.3: Structure for handwritten numeral recognition with the CNN network



Numeral Recognition using TensorFlow To Program ANN

Example 8.3: Use of Convolutional Neural Network for Numeral Recognition

For the same purpose as in the last two Examples, we want to recognize the handwritten numerals using the Mnist dataset and CNN construction.

Step 1：Load data set

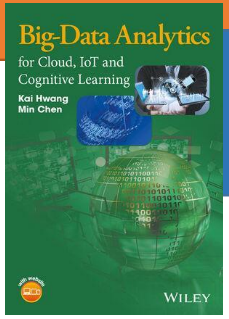
First, we need to load data set and preprocess the train set train_x and label set train_y, respectively. Similarly, we also need to preprocess test data set test_x and label set test_y. The Pseudo-code of these tests is given below:

```
load mnist_uint8;
train_x = double(reshape(train_x',28,28,50000))/255;
test_x = double(reshape(test_x',28,28,10000))/255;
train_y = double(train_y');
test_y = double(test_y');
```

Step 2：Initialization in CNN

In this step, we need to initialize the structure of convolution layer and pooling layer. For convolution layer, the number of convolution kernel (outputmaps) and the size of convolution kernel (layerCkernel). For pooling layer, the size of pooling area (layerScale) should be specified. The pseudo-code is given below

```
layerNumber=5
layer(1，'i')               //  Input layer
layer(2, 'c', 6, 5)         // Convolution layer,  6 5x5  kernel,
            layerCkernel=5, layerName='c'，outputmaps=6
layer (3, 'p', 2)           //  Pooling layer, 2x2 Pooling, layerScale=2,
            layerName='p'
layer (4, 'c', 12, 5)       //  Convolution layer, 12 5x5 kernel,
            layerCkernel=5, layerName='c'，outputmaps=12
layer (5, 'p',2)            //Pooling layer, 2 x 2 Pooling, layerScale=2,
            layerName='p'
```



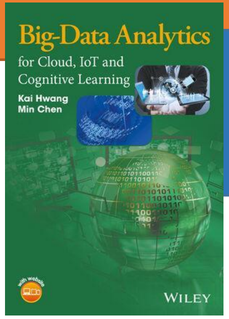
Numeral Recognition using TensorFlow To Program ANN

For the Msame purpose as in the last two Examples, we want to recognize the handwritten numerals using mnist dataset and CNN construction.

Step 3：Training CNN
During the Training stage, we first input the handwritten numeral images. Output classes of input images will be obtained after forward propagating algorithm goes through all of layers at CNN.

(1) CNN forward propagating
(2) CNN backward propagating
(3) Update weights

Step 4：Test CNN
The final testing requires the input test_x and utilize cnnff() to output the recognition results. The testing tag set test_y is used for testing the accuracy of the recognition process.



Convolutional Neural Networks for Face Recognition

- Human face recognition is a very important research direction in computer vision field. At present, in the field of human face recognition, deep learning has reached or exceeded human level.

- The structure of CNN adopted in Deep ID algorithm, proposed by The Chinese University of Hong Kong in 2014, whose purpose is human face recognition. The Deep ID algorithm includes 4 convolution layers, 3 pooling layers and 1 full connection layer. The recognition rate for the human face in the LFW data set with such a CNN achieves 87.45%.

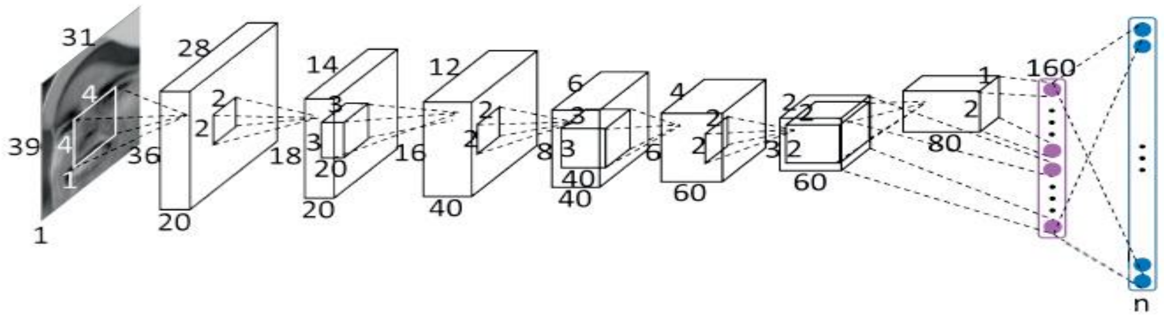
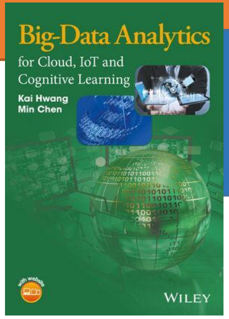


Figure 8.4: Structure of Deep ID human face recognition system with a CNN with 10 layers including the I/O layers.



Word Embedding via Deep Learning

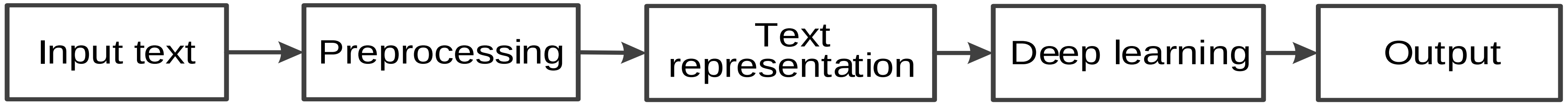
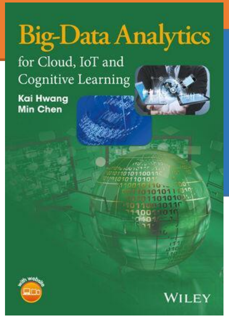


Figure 8.5: The process of text classification with deep learning

Generally, we adopt the method of word embedding for representation. There are two representation methods of word embedding: one-hot representation and distributed representation. One-hot representation is simpler and more direct. The quantity of total words in vocabulary is also the dimension of vectors.

But there is only one value equal to 1 in vector composition of each word while other values are equal to 0. There is no relation between the vectors even for words with similar semanteme, which is called a “word gap”. As the dimension of word embedding is equal to quantity of total words, the computational burden for application in some tasks is too high and it may cause a dimension disaster.



Word Embedding via Deep Learning

Figure 8.6 shows a text representation with one-hot representation method. Thus the dimension of vectors is much less than the quantity of total words. Tool Word2vec2 is often used to train word embedding. The dimension could be set to 50. With this method, the word embedding introduces semanteme of words, i.e., the closeness among words makes their word vectors closer in the vector space.

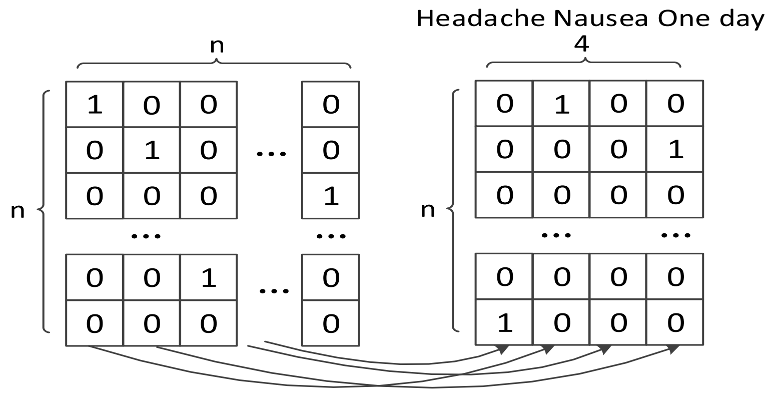
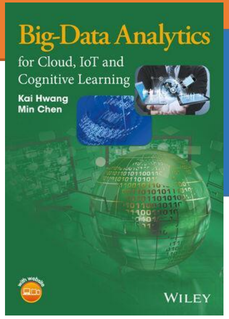


Figure 8.6 One-hot representation for word embedding



Word Embedding via Deep Learning

Figure 8.7 shows a text representation with a distributed representation method. Compared with one-hot representation, the dimension of word embedding is largely reduced and the distance of vectors between relevant semanteme or similar semanteme is close.

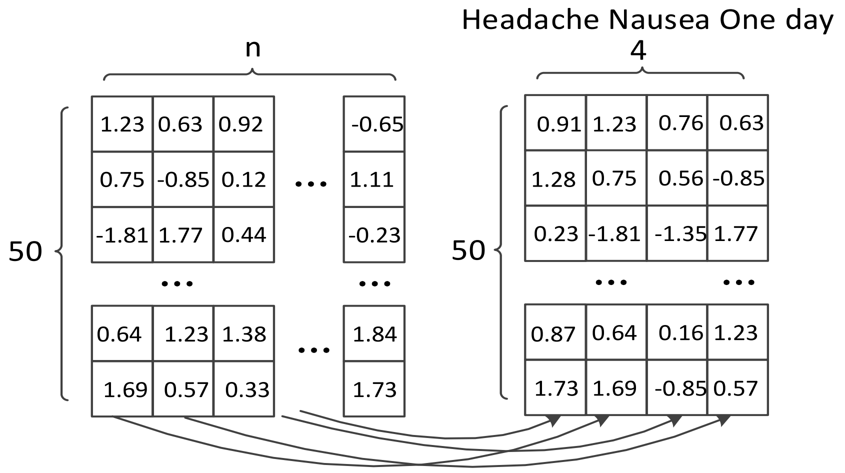
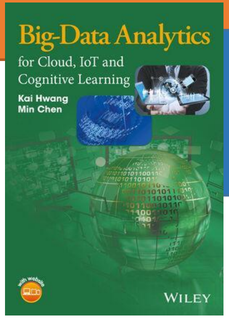


Figure 8.7 : Distributed representation for word embedding



Word Embedding via Deep Learning

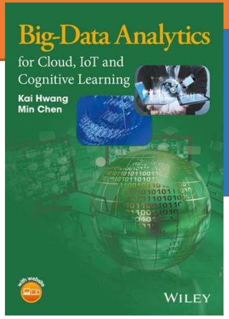
- Word Embedding is a matrix: $\mathrm{D}=R^{d\times|\mathrm{C}|}$

  where *d* is the dimension for word embedding. |C| is quantity of words in vocabulary. The locations of words in vocabulary are stored in C.

- Each input text sample x(x1，x2，…，xN) includes N words. We search the vector representation xwn corresponding to each word xn in a text from word embedding by:

$$\mathrm{t}=\mathrm{D}\cdot\mathrm{C}$$

The word embedding representation xw(xw1，xw2，…，xwN) will be obtained for input sample.



Medical Text Analytics using the CNN

- As shown in Fig. 8.8, we show a model of medical text comprehension based on convolutional neural network.

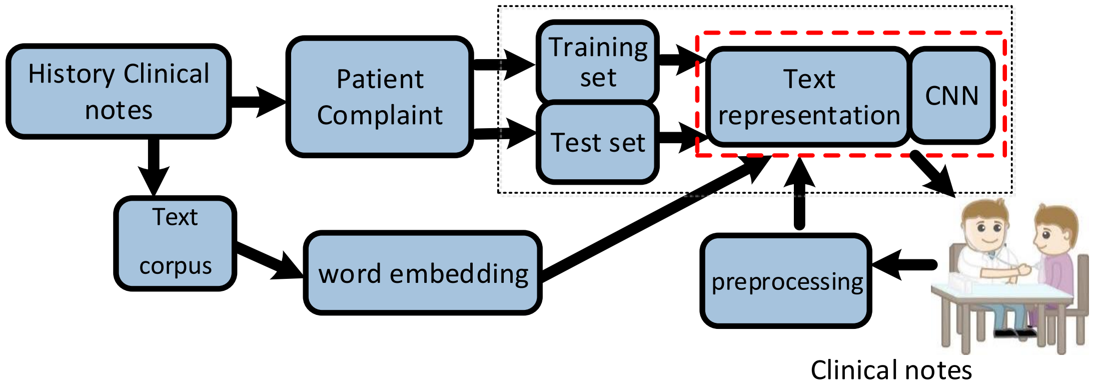
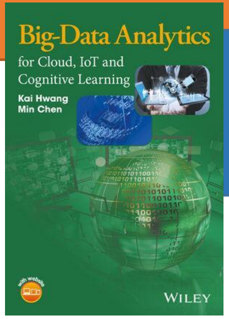


Figure 8.8: Model for risk disease assessment through medical text learning using CNN network



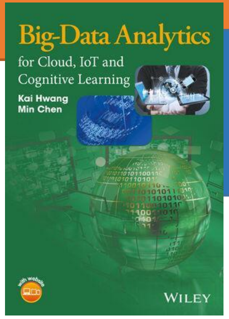
Medical Text Analytics using the CNN

The model mainly includes three parts:

- Setup Word Embedding: it extracts data of a disease from data in clinical notes. After data cleaning and data preprocessing, it selects “Patient Complaint” and “Diagnosis Record through Interrogation”, etc. It makes the processed data as sample data, conducts digitized representation for sample data with word embedding and inputs the results into CNN for supervised learning of features in risk assessment of diseases.

- Train CNN to learn features of medical text: It chooses data of a disease from data in clinical notes; after data cleaning and data preprocessing, it select “Patient Complaint”, “Diagnosis Record through Interrogation” and etc. And the data becomes sample data. Then, conduct digitized representation for sample data with word embedding, and input the results into CNN for supervised learning of features in risk assessment of diseases.

- Test and application: The processes of test and application are the same. It inputs “Patient Complaint” and disease-relevant text data, conducts preprocessing and text representation for data, inputs results into CNN and outputs the result of risk assessment of diseases.

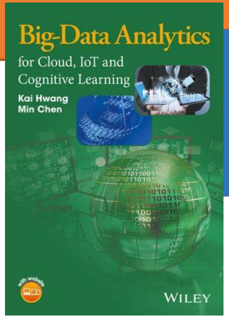


Convolutional Neural Network Construction

For word embedding representation xw (xw1, xw2, …, xwN) of input text, it calculates the convolution vector s_n^{sc} for each word in xw in sequence. The calculation of convolution vector for word n is given in Fig. 8.9. The size of convolution window is d^s . With current word n as center, intercept words from the sequence of input text and connect the vectors of these words.

$$S_n \in R^{dd^s}$$

$$S_n = (xw_{n-(d^s-1)/2}, \ldots, xw_n, \ldots xw_{n+(d^s-1)/2})^T$$



Convolutional Neural Network Construction

It calculates the convolution vector s_n^{sc} for word *n* using the following equation, where, $w^1 \in R^{l^{sc} \times dd^s}$ is weight matrix and b^1 is deviation.

$$s_n^{sc} = w^1 s_n + b^1$$

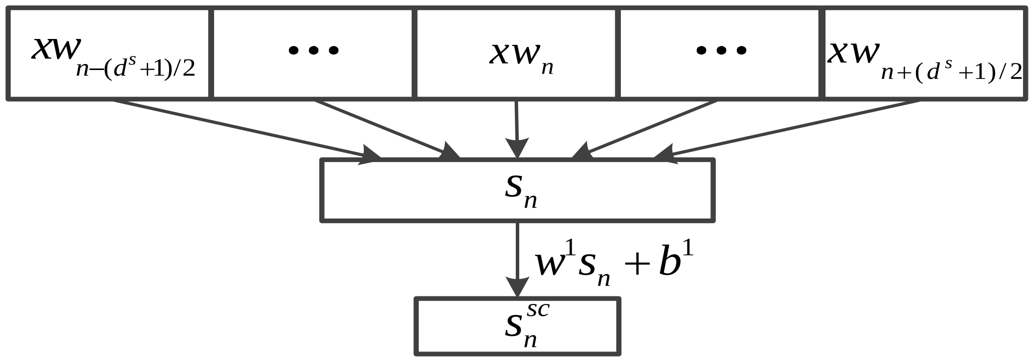
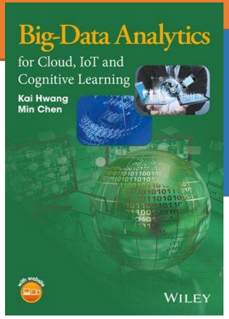


Figure 8.9: Convolution for word xw_n (word No. n)



Convolutional Neural Network Construction

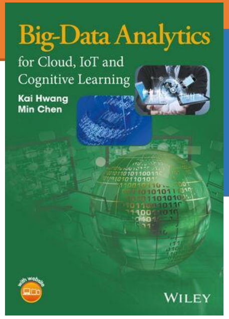
The variable s_n^{sc} stands for h_n^1 (the expression of word xwn in hidden layer). After obtaining h_n , we calculate the output h_n^2 of hidden layer with tanh-function ($\tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}}$) and make it as the input of the next hidden layer.

$$\mathrm{h}_n^2 = \tanh(w^1 s_n + b^1)$$

- Pooling layer: The output of convolution layer is used as the input of pooling layer. The maximum values among N elements in h_n^2 is calculated by:

$$\mathrm{h}^3 = max_{1 \le n \le N}\, \mathrm{h}_n^2$$

Pooling operation is divided into maximum pooling operation and average pooling operation. Here we conduct maximum pooling operation because the function of each word in text is not completely equal. Text will be transformed into vector with fixed length after it goes through convolution layer and pooling layer.



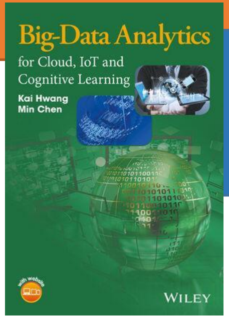
Convolutional Neural Network Construction

- Output layer: There is a full connection layer of neural network after pooling layer. Softmax classifier is adopted to output classification results.

$$\mathrm{h}^4 = w^4\,\mathrm{h}^3 + b^4 \rightarrow y_i = \frac{e^{\mathrm{h}_i^4}}{\sum_{m=1}^{n} e^{\mathrm{h}_m^4}}$$

where *n* shows there are *n* classes. We define all parameters that require training as parameter set $\theta = \{w^1, w^4, b^1, b^4\}$. The objective of training is to learn maximum log-likelihood value of *θ*. We utilize random numbers as initialization parameter of *θ* and stochastic gradient descent for training of parameter *θ*. As for parameter revision, we utilize the following expression, D is training sample set, $class_y$ is accurate classification of sample, and *α* is the learning rate.

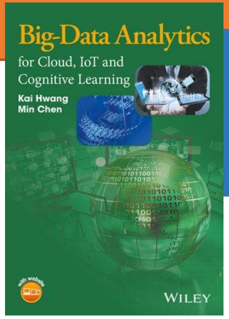
$$\max_{\theta} \sum_{y\in \mathrm{D}} \log p\left(class_y | \mathrm{D}, \theta\right) \rightarrow \theta = \theta + \alpha \frac{\partial \log p\left(class_y | \mathrm{D}, \theta\right)}{\partial \theta}$$



Medical text comprehension with CNN network

The pseudo-code for disease risk assessment model in medical text comprehension with CNN network is specified in Algorithm 8.1. Medical text data include training set X and test set X', corresponding results data of diagnosis from doctors are divided into training label set Y and test label set Y'. The training set is represented by X and corresponding label set by Y.

First we read one medical text data x from X, then represent it as a vector, finally obtaining the predicted result y* after convolution layer, pooling layer and fully connection layer. The update of parameter in this CNN is completed through backward propagation algorithm. The CNN after training can be used do risk assessment on disease according to medical text. In the same way, we use the test set X' and corresponding label set Y' to test the model. We get the predicted result y* and contrast it with the true label in Y' to estimate the performance of the CNN.

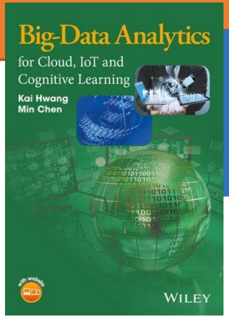


Medical Text Comprehension with CNN Network

Algorithm 8.1 Creation of CNN for Text Recognition

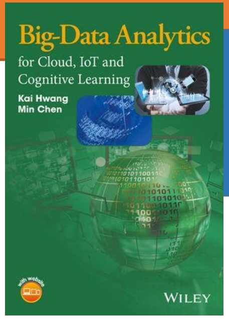
Input: X: Training sample, original input data in clinical notes
Y: Tag for training sample, corresponding disease diagnosis result of patient in clinical notes
Output: Create CNN, network parameter $\theta = \{w^1, w^4, b^1, b^4\}$, test result
Algorithm:

```
1)  Initialization: c = 5 // the size of convolution kernel;
                    T = 50 // the size of sample batch
2)  for $I$ = 1,2,…, I // (I is the number of iterations)
3)  for j = 1,2,…, m // (m is the number of sample batches)
4)  Read in a sample batch x and corresponding tag y
5)  Vector representation for data x (as for each word, search its vector representation in word embedding).
6)  for n = 1,2,…, T
7)  Work out number of words (n) in sample
8)  Calculate convolution
9)  Max-Pooling for n words
10) Connect full connection layerwith softmax classifier for classification, get result$y_n^*$ .
11) End
12) Use the backward propagation algorithm to update $\theta$ with Formula 10.5
13) End
```



Google DeepMind AI Programs

- In 1997, IBM's computer Deep Blue beat world chess champion Garry Kasparov in an open match.
- The AlphaGo research project was formed around 2014 to test how well a neural network using deep learning can win over Go professional players.
- The AlphaGo and Lee match proves that computers can be trained to formalize human intelligence processing.
- Google DeepMind has combined deep learning and reinforcement algorithm to achieve human level performance in several innovative AI applications.



Google DeepMind AI Programs

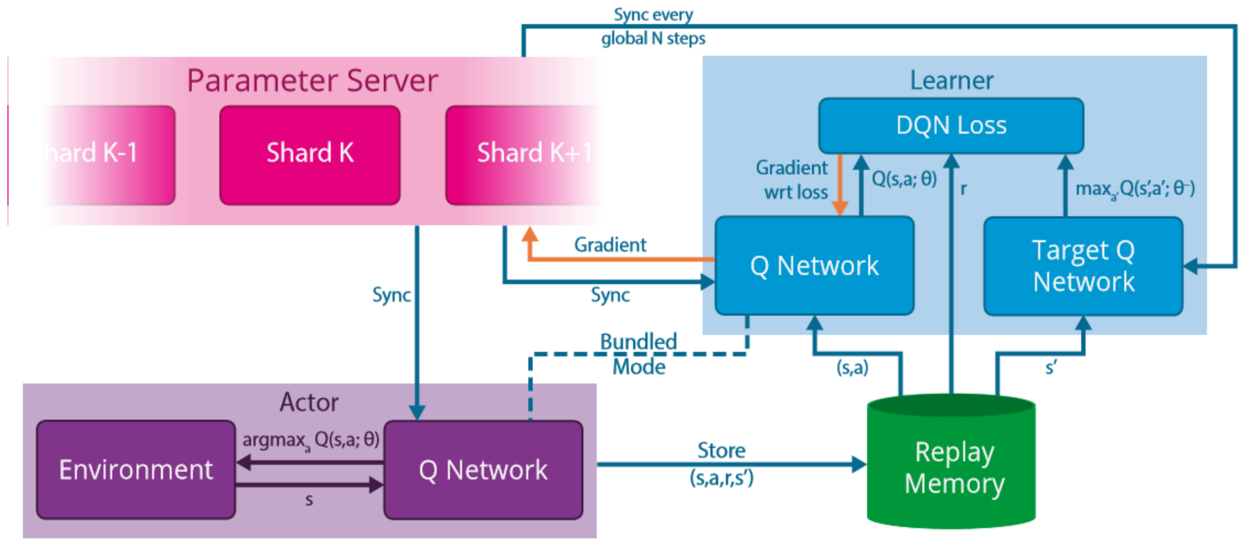
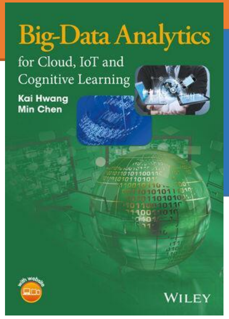


Figure 8.10: The Glorila architecture for implementing the Google reinforcement learning system

(Courtesy David Silver, Google DeepMind, http://www0.cs.ucl.ac.uk/staff/d.silver/web/Resources_files/deep_rl.pdf)

In what follows, we introduce the DeepMind approach by combining deep learning with reinforcement learning ideas. Then we examine the algorithms used in AlphGo and in Floppybird game, including their implementation and learning process applied.



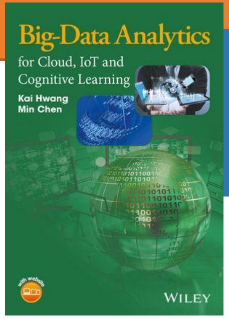
Deep Reinforcement Learning Algorithm

- As introduced in Section 8.1.2, the RDL process is mainly displayed by interactions between the learning agents and its working environment. Without knowing any rules in advance, an agent observes the current environmental state and tries some actions to improve the deep learning process.

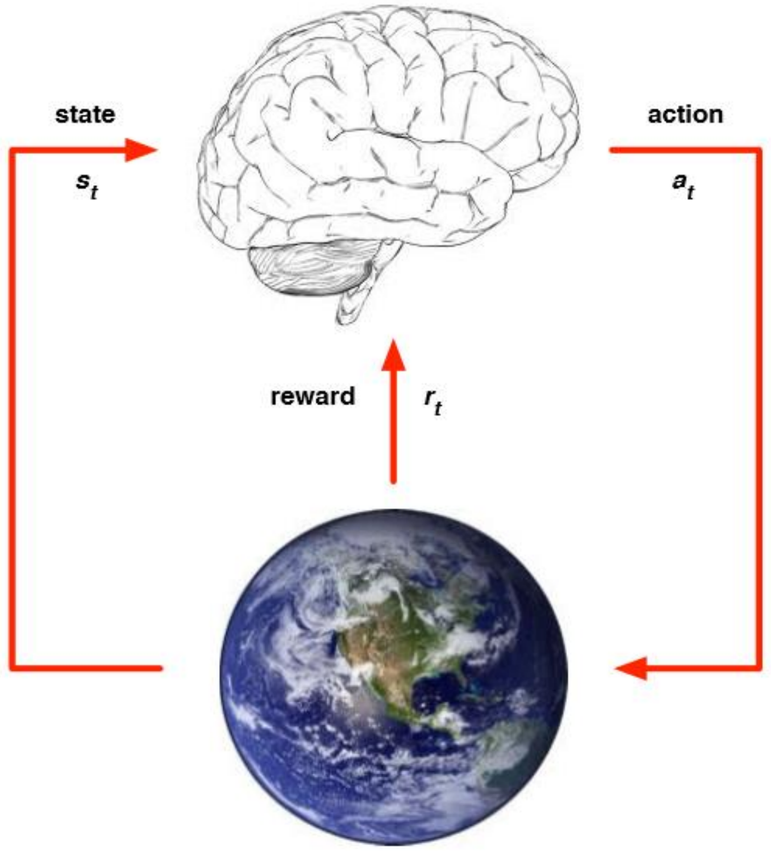
- A reward is the feedback to the agent by adjusting its action strategy. After numerous adjustments, the reinforcement algorithm obtains the knowledge of optimal actions to achieve best results for a specific situation in the decision environment.

- Figure 8.11 shows the interaction of an agent and its environment during the learning process. At each time t, the agent receives a state s_t and executes an action a_t. Then, it receives an observation o_t. and a reward r_t associated with the action.

- The goal of reinforcement learning is to accumulate the rewards as much as possible at successive steps. A sequence of observation, action and reward, $\{ o_1, r_1, a_1, \cdots, a_{t-1}, o_t, r_t \}$, forms an experience, while the state is a function of the experience, i.e.

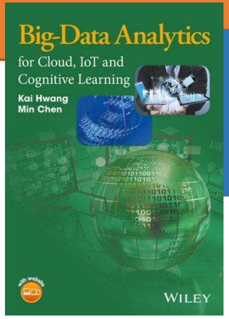


Deep Reinforcement Learning Algorithm



- At each step t the agent:
  - Receives state s_t
  - Receives scalar reward r_t
  - Executes action a_t
- The environment:
  - Receives action a_t
  - Emits state s_t
  - Emits scalar reward r_t

Figure 8.11 : Interaction of an agent and environment in deep learning (From the speech of David Silver in ICML 2016)



Deep Reinforcement Learning Algorithm

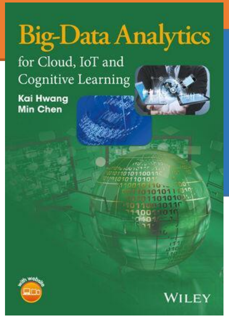
In the 33rd International Conference on Machine Learning (ICML 2016), Silver *et al*. presented the application details of DeepMind by using the DRL approach. After the action is selected by the agent, policy and value function play important roles for its performance.

- **Policy** is a behavior function selecting actions given states.
- **Value function** predicts the future reward, and evaluate the efficacy of an action or state.

$$Q^{\pi}(a \mid s) = E[r_{t+1} + \gamma r_{t+2} + \gamma^{2} t_{t+3} + \cdots \mid s, a]$$

The goal is to obtain the maximum value of $Q^{\pi}(s,a)$. The optimal policy is obtained by maximizing the value function as follows.

$$Q^{*}(s,a) = E[r_{t+1} + \gamma \max_{a_{t+1}} Q^{*}(s_{t+1}, a_{t+1}) \mid s, a]$$



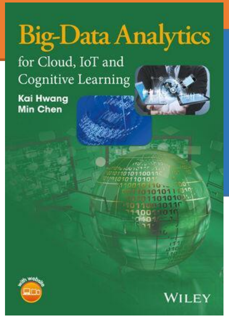
Deep Reinforcement Learning Algorithm

Given a transit < *s*, *a*, *r*, *s′* >, the Q value table updating algorithm is given below:

1) Perform forward propagation starting from current state s and obtain predicted Q value for all actions;
2) Perform forward propagation starting from the next state s', calculate the maximum Q value, i.e., $\max Q(s',a')$;
3) Set the target Q value $r+\gamma\max Q(s',a')$ based on the calculated result at Step 2 and the predicted value $Q(s,a)$ calculated by step 1.
4) Update weight *s* of the neural network by the backward propagation algorithm. The loss function is shown as

$$L=\frac{1}{2}[\underbrace{r+\max Q(s',a')}_{target}-\underbrace{Q(s,a)}_{prediction}]^2$$

5）The model refers to a descriptive expression to specify the agent behavior in the learning environment. As a result, the agent must be able to take appropriate action strategies based on the current environment and the past experience, in view of the fact that the environment is unknown or under constant change.



CNN Construction in AlphaGo and Its Training Process

The Go game is played on a 19x19 grid board as shown in Fig.8.12. Black and white stones are place on the board, one at a time by two players alternately.

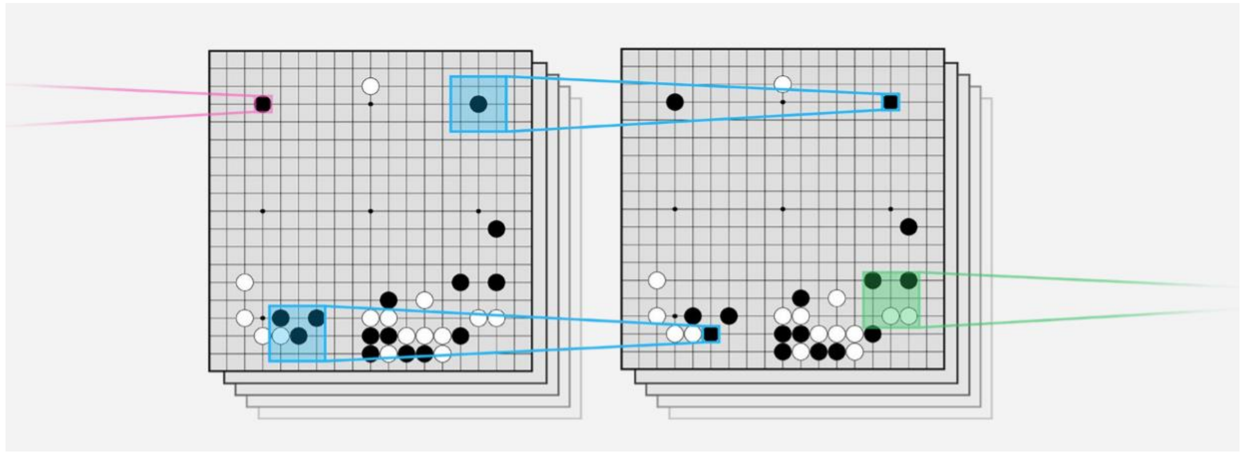
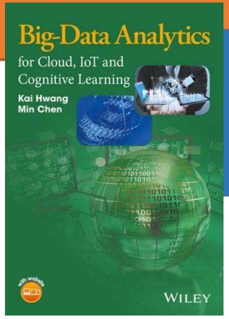


Figure 8.12 Convolutional neural network construction over the Go playing board.



CNN Construction in AlphaGo and Its Training Process

Figure 8.13 illustrates the neural network training process using human expert positions. This process is repeated in many iterations until the winning condition is met. The supervised learning and reinforcement learning are explained in Example 8. 4.

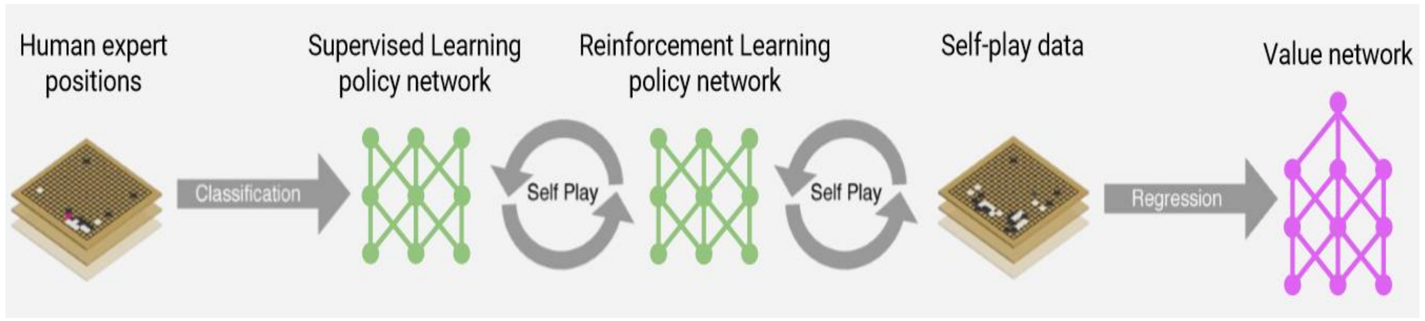
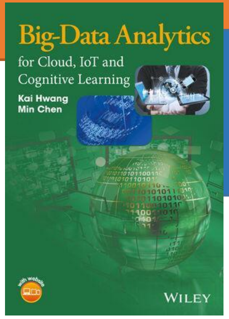


Figure 8.13 The self-play training pipeline between policy network and value networks by human experts. (Courtesy of Google DeepMind, http://icml.cc/2016/tutorials/AlphaGo-tutorial-slides.pdf)



CNN Construction in AlphaGo and Its Training Process

Example 8.4 : suprevised and Reinforcement learning of policy networks before feeding the self-play data into the value network using regression

This example shows the specific training details of supervised learning followed by reinforcement learning on the policy networks.

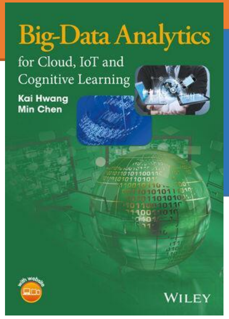
$$\Delta\sigma \propto \frac{\partial \log \mathrm{p}_{\sigma}\left(a \mid s\right)}{\partial\sigma} z$$

where s is the state, a the action, and σ the reward.

Its training algorithm minimizes the mean square error θ by a stochastic gradient descent method characterized by:

$$\Delta\theta \propto \frac{\partial v_{\theta}\left(s\right)}{\partial\theta}\left(z - v_{\theta}\left(s\right)\right)$$

The entire pipelined of policy and value networks can achieve a first strong position evaluation that has never been achieved previously by other Go programs.



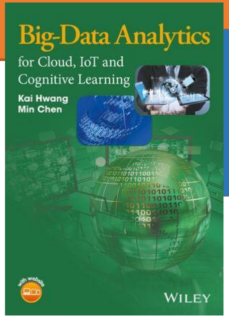
System Architecture for Deep Reinforcement Learning in AlphaGo Program

In Fig. 8.14, we provide a schematic block diagram to illustrate the data flow in the AlphaGo program in three learning stages. The 3 stages are described below.

- **Stage 1:** This stage performs off-line deep learning as shown on the left side .
  The purpose is to perform two tasks in parallel: (1) Local feature extraction with a linear model training to generate fast rollout for use in the Monte Carlo Tree search (Fig. 8.15). (2) Run feature graph in 48 passes with a deal learning model to update the policy network .

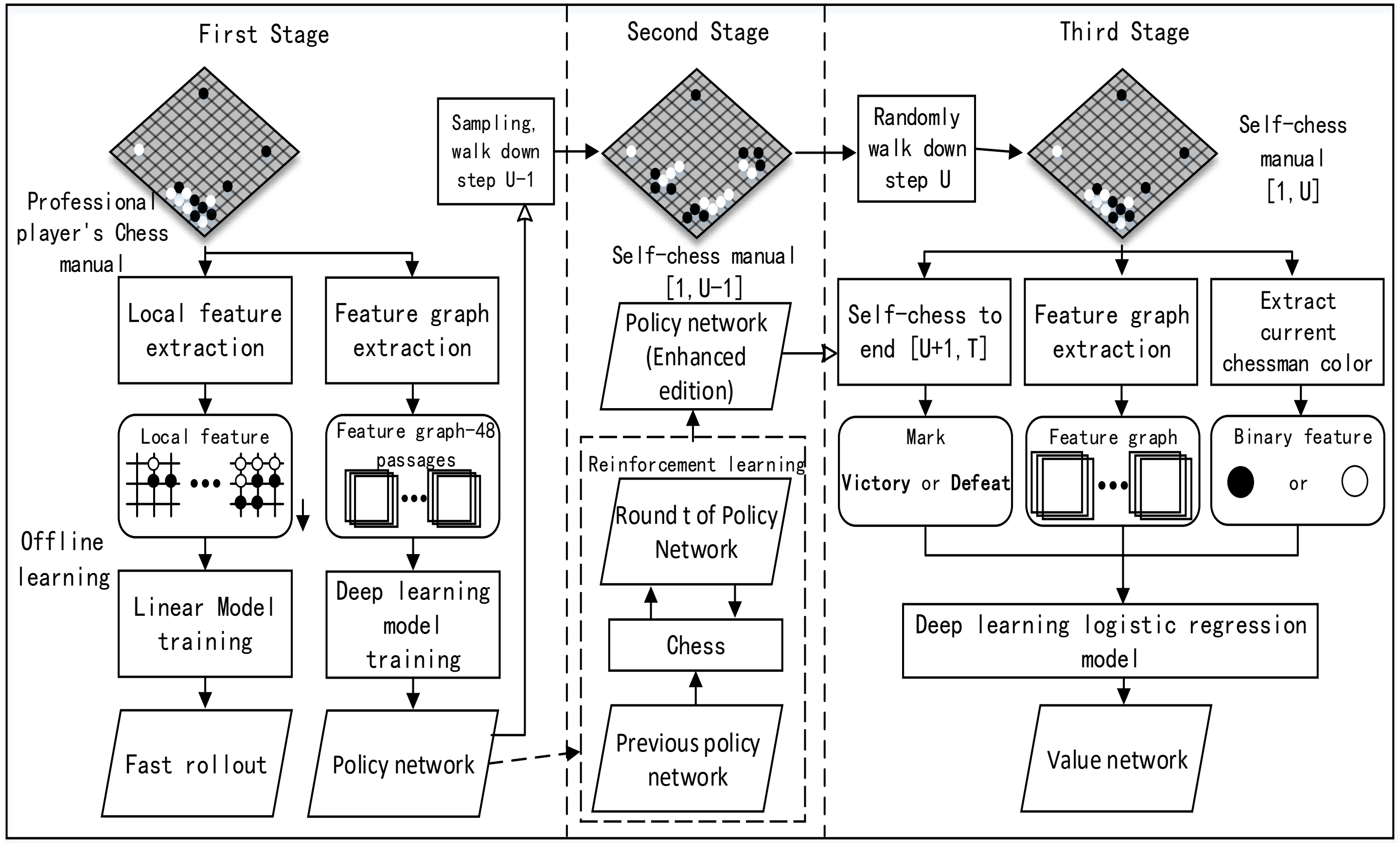
- **Stage 2:** This stage updates the previous policy network through reinforcement learning to an enhanced edition of the policy network.

- **Stage 3:** This stage applies the self chess manual at step u in three parallel tasks for mark victory or defeat, extract useful features, and current chessman color.



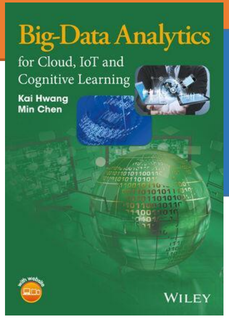
System Architecture for Deep Reinforcement Learning in AlphaGo Program

The outputs of the 3 tasks are merged to feed into deep learning logistic regression model in order to work with value network. The updated rollout, policy network and value network will be used in the on-line execution process in 5 steps specified in Fig. 8.15.



First Stage
Second Stage
Third Stage
Professional player's Chess manual
Sampling, walk down step U-1
Local feature extraction
Feature graph extraction
Local feature
Feature graph-48 passages
Offline learning
Linear Model training
Deep learning model training
Fast rollout
Policy network
Self-chess manual [1, U-1]
Policy network (Enhanced edition)
Reinforcement learning
Round t of Policy Network
Chess
Previous policy network
Randomly walk down step U
Self-chess manual [1, U]
Self-chess to end [U+1, T]
Feature graph extraction
Extract current chessman color
Mark
Victory or Defeat
Feature graph
Binary feature
or
Deep learning logistic regression model
Value network

Figure 8.14: The off-line learning process of the AlphaGo program (Courtesy of art work by Yu Zeng, Huazhong University of Science and Technology, China)



Execution Steps in Online AlphaGo Match with a Human Player

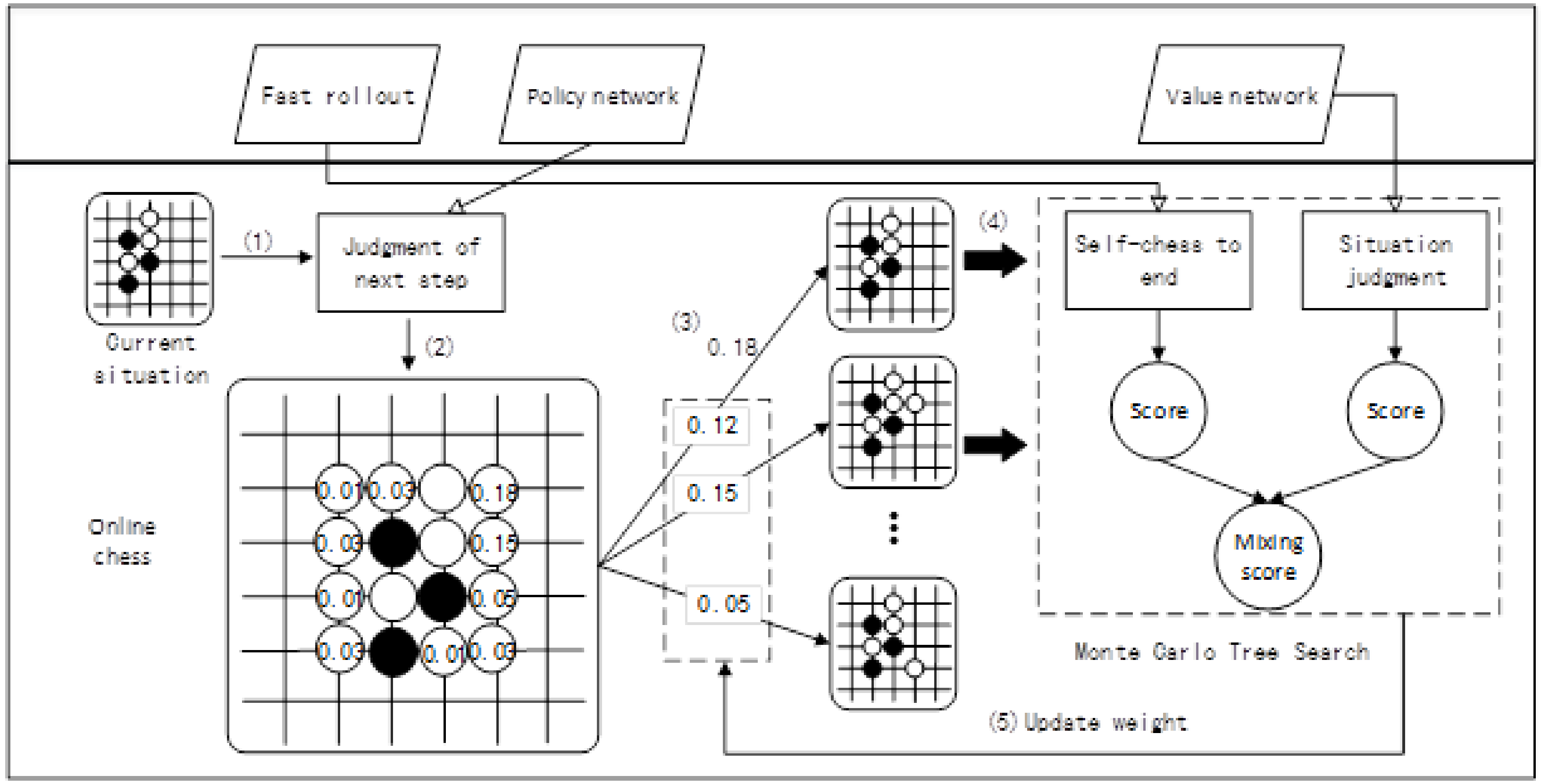
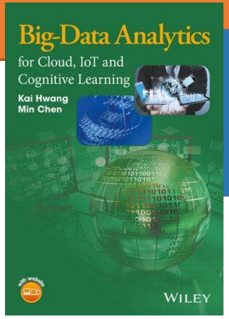


Figure 8.15: The on-line playing process of the AlphaGo game (Courtesy of art work by Junbo Zhang, Huazhong University of Science and Technology, China)



Execution Steps in Online AlphaGo Match with a Human Player

The MCTS performs the following tasks using the value network and policy network, collaboratively:

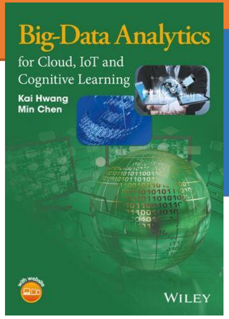
(1) Choose several possible strategies that the opponent would choose for their next move based on current situation.

(2) Judging the strategies of the opponent, choose a move that benefits most to traverse the right subtree. The search tree of AlphaGo will not expand all the nodes, except along the optimal path of the subtree traversed.

(3) How to estimate the best action in the next move requires to estimate the probability of winning with the value network. A Monte Carlo tree search needs to predict deeper results along the tree layers. The mutually supportive results of these two networks are key for AlphaGo to win the game.

(4) After deciding the best action, we estimate the possible next moves of opponent and corresponding strategies through policy network based on the location of best action.

In summary, the reinforcement learning method is based on the Monte Carlo search tree on the on value network and the policy network, which are both implemented with deep neural networks.

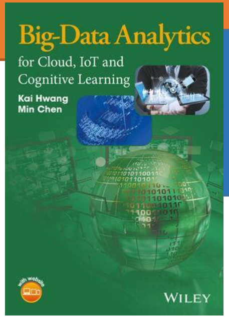


Execution Steps in Online AlphaGo Match with a Human Player

Example 8.5: Reported Performance Results on Monte-Carlo Tree Search

The MCTS process essentially exhausts all possible moves and rewards. Building a large lookahead search tree to cover millions of possibilities, the 19 $\times$ 19 AlphaGo program uses MCTS to end up with high accuracy. The value network is trained to predict expert human moves, using the large database of professional Go games. Detailed performance data can be found in David Silver, Google DeepMind, http://www0.cs.ucl.ac.uk/staff/d.silver/web/Resources_fles/deep_rl.pdf

The predictive accuracy of the 12-layer CNN reaches 55%, which is a significant improvement over the 31% and 39% predictive accuracy reported for earlier Go programs. The neural network is considerably stronger than the traditional search-based program GnuGo, and its performance is on a par with MoGo for 100,000 rollouts per move. The Pachi runs a reduced search of 10,000 rollouts per move. It wins about 11% of games against Pachi, with 100,000 rollouts per move.



Flappybird Game using Reinforcement Learning

One of DQN's representative applications is playing Atari 2006, a collection of popular entertainment computer games. It includes 49 independent games, such as Breakout and other classic games. The input of the algorithm consists of the image of game screen and the game score. Without knowing the rules of the games, the DQN learns how to play the games by itself and find the best strategy to play.

Figure 8.17 shows the Artari game setting involving the joystick and the slide control to play the game. The interplay among the learning state, action, and reward is shown by the arrows.

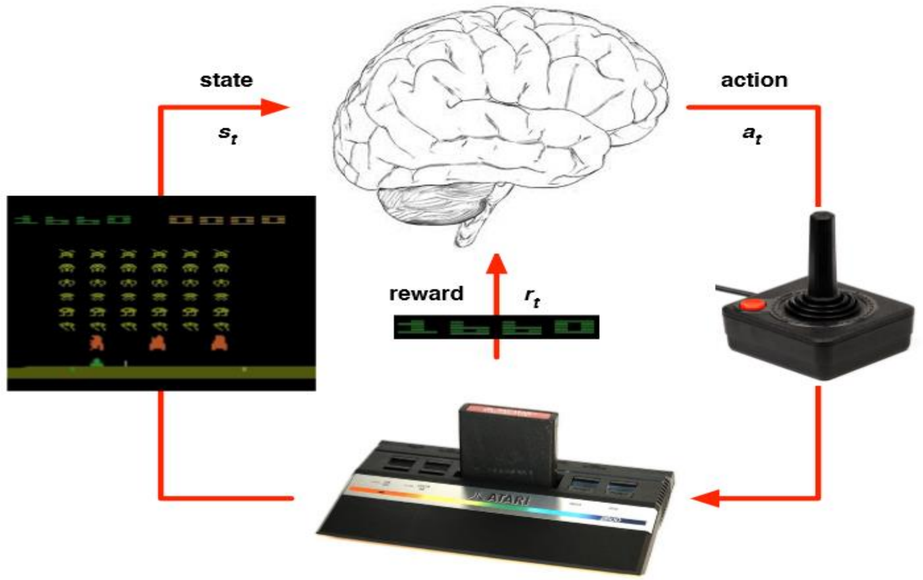
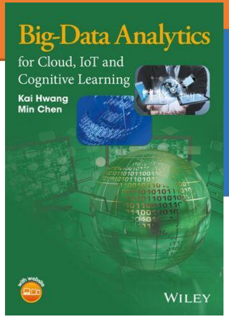


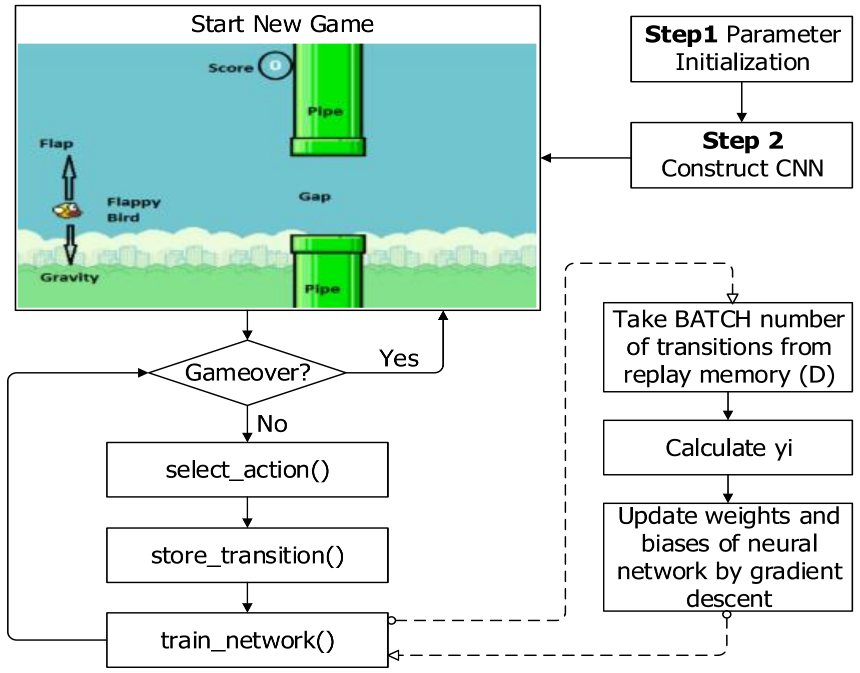
Figure 8.16 Reinforcement learning applied in Artari game play



Flappybird Game using Reinforcement Learning

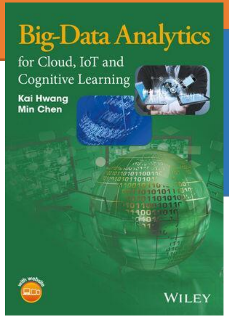
Example 8.6: Deep Q Network works the Neural Network in Artari Games

Flappybird is a simple game based on using the DQN, as illustrated by the flowchart in Fig. 8.17. The player must control a bird not to fly too high or too low to hit the water pipes.



Start New Game
Score 0
Pipe
Flap
Flappy Bird
Gap
Gravity
Pipe
Step1 Parameter Initialization
Step 2 Construct CNN
Gameover?
Yes
No
select_action()
store_transition()
train_network()
Take BATCH number of transitions from replay memory (D)
Calculate yi
Update weights and biases of neural network by gradient descent

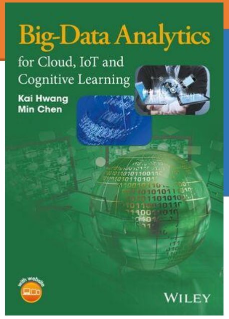
Figure 8.17: The flowchart of DQN algorithm for playing the Flappybird game



Flappybird Game using Reinforcement Learning

This CNN includes 3 convolution layers and 1 full connection layers. In the game, there are two actions that the player can take: to press the 'up' key, which makes the bird jump upward or not pressing any key, which makes it descend at a constant rate. The pseudo code for creating the CNN is given below:

```
def createNetwork():
    # Weight of neural network
    # Input layer
    # Hidden layer
  # Output layer
        Qvalue= tf.matmul(h_fc1, W_fc2) + b_fc2         // Predict the value of Q
     return s, Qvalue, h_fc1
     a = tf.placeholder("float", [None, ACTIONS])        // Allocating space to action
     y = tf.placeholder("float", [None])                 // y is the Q value of optimal goal
      Qvalue_action = tf.reduce_sum(tf.mul(Qvalue, a), reduction_indices=1)
                                                         //Neural network prediction value
      cost = tf.reduce_mean(tf.square(y - Qvalue_action))          // Loss function
      train_step = tf.train.AdamOptimizer(1e-6).minimize(cost)
                          //Neural network optimization by minimizing the loss function
```



Flappybird Game using Reinforcement Learning

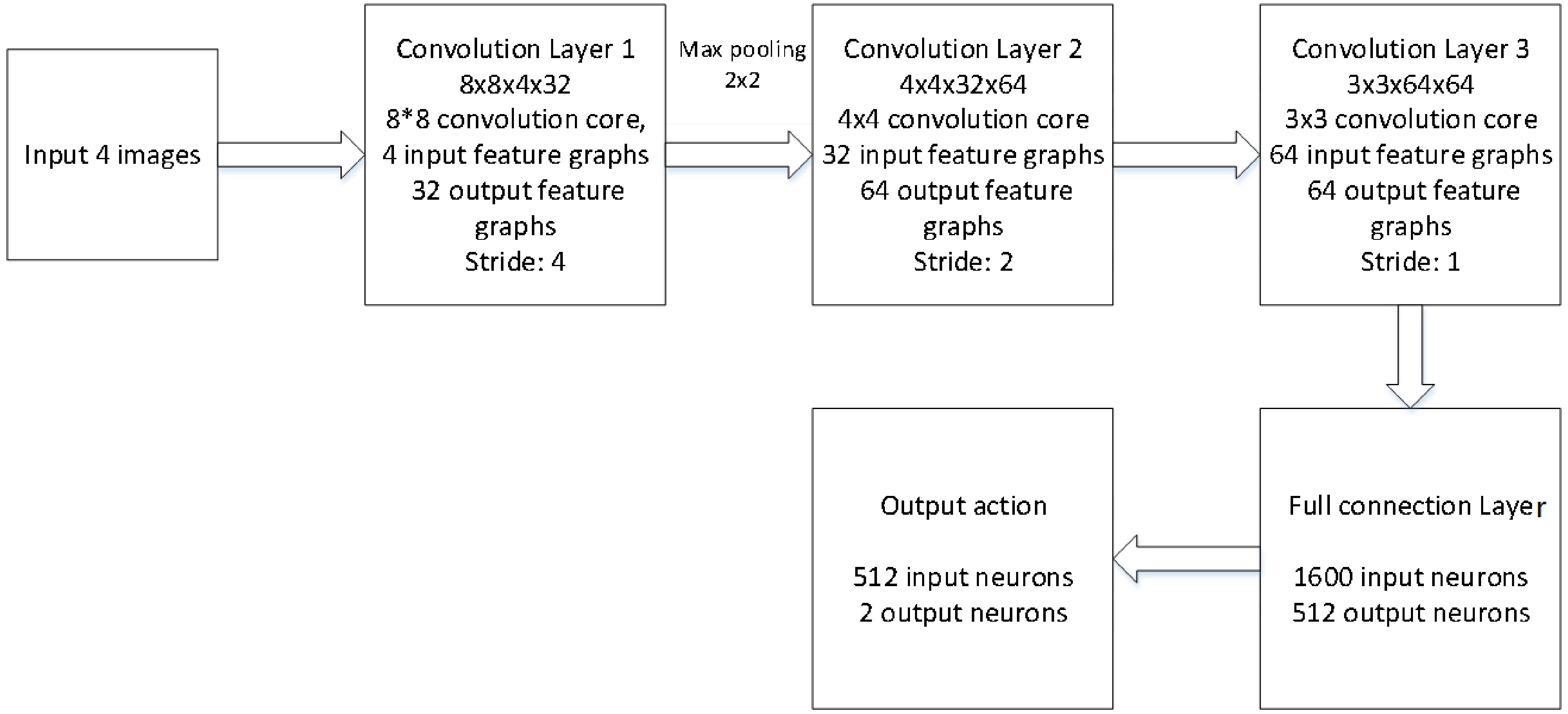
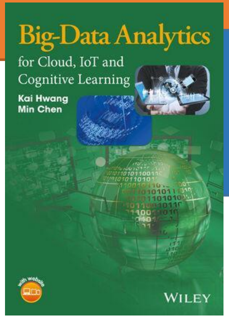


Figure 8.18: The construction of the convolutional neural network used in FlappyBird Game

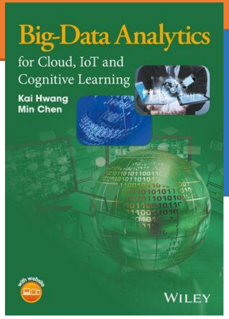


Flappybird Game using Reinforcement Learning

The samples are stored in replay memory with the form of a sequence $(\phi_t, a_t, r_t, \phi_{t+1})$ and then extract BATCH (the number of minibatches taken out from the replay memory each time) sequences randomly to conduct training. In most cases, the game scenes can be captured in a very short time while feature coding by DL and policy building by RL are computing-intensive with a longer delay. Thus, when every new frame appears during game playing, it needs to check whether the agent finishes the calculation.

Every four frames of game screen together as a training sample. After obtaining a new image, the next state will move forward one frame to ensure that the image is still 4 frames. Then, the training sample will get the operation set. As for each sample, we should initialize stats s_1 first. Now begin the game. The pseudo-code for selecting the actions is given below.

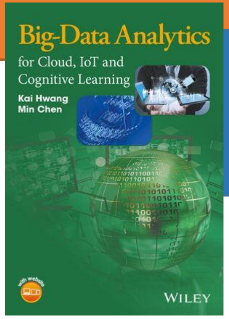
```
def select_action():
Qvalue_t = Qvalue.eval(feed_dict={s : [s_t]})[0]        // get Qvalue_t through neural
network
        a_t = np.zeros([ACTIONS])          // allocate space for action
        action_index = 0          // index for action
if  t % FRAME_PER_ACTION == 0:          // each action skips on frame of samples
         if  random.random() <= epsilon:      //randomly choose one action
                   action_index = random.randrange(ACTIONS)
                   a_t [random.randrange(ACTIONS)] = 1
         else:             // take action based on the best policy
                   action_index = np.argmax(Qvalue_t)        // predict all the Q values for
                                 each action, and choose the action with maximum Q value
                   a_t [action_index] = 1
else:      a_t [0] = 1       //do nothing
```



Flappybird Game using Reinforcement Learning

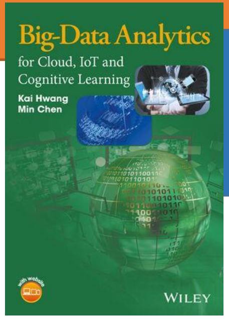
The training of the deep CNN is performed **by the following pseudo code.**

```
def train_network():
if  t > OBSERVE:
     minibatch = random.sample(D, BATCH)          //choose BATCH sequences from D
    s_j_batch = [d[0] for d in minibatch]          //current state of corresponding sequence
    a_batch = [d[1] for d in minibatch]            //action of corresponding sequence
    r_batch = [d[2] for d in minibatch]            //reward of corresponding sequence
    s_j1_batch = [d[3] for d in minibatch]         //next state of corresponding sequence
    y_batch = []                                   //allocate space for y
    Qvalue_j1_batch = Qvalue.eval(feed_dict = {s : s_j1_batch})
                                        // compute the max Q value for the next state
    for  i in range(0, len(minibatch)):
            terminal = minibatch[i][4]             //next state of sequence
            if  terminal:
                  y_batch.append(r_batch[i])
            else:
            y_batch.append(r_batch[i] + GAMMA × np.max(Qvalue_j1_batch[i]))
            train_step.run(feed_dict = {
            y : y_batch,
            a : a_batch,
            s : s_j_batch})
```



Data Analytics for Social-Media Applications

- Social media is a major source of big data aggregation in our daily activities.
- In this section, we assess data analytics technologies applied in the social-media industry and its impact on all walks of life. Then we study social networks and graph analysis of social communities. Finally, we present smart cloud resources needed to support big-data analytics applications.

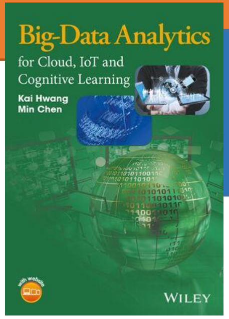


Big Data Requirements in Social-Media Applications

- Mobile social-media users make use of the location and/or time sensitive features of the big dataset collected.

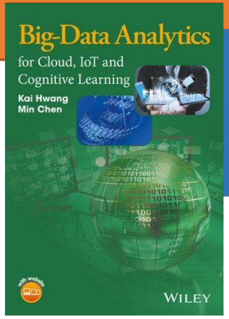
  We can assess in the four areas.

  - Marketing research
  - Communication in social-media exchanges
  - Sales promotions and discounts
  - e-Commerce



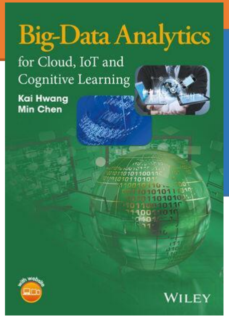
Big Data Requirements in Social-Media Applications

- Marketing research
  - In mobile social media applications, the users often collect data from offline consumer movements first, before they move to online companies. On-line data collections could escalate rapidly to a large amount. Handling timely in a streaming mode continuously, the requirement is to keep all concerned parties or firms well informed with the exact times of transactions and the comments made during the transactions or social network visits.



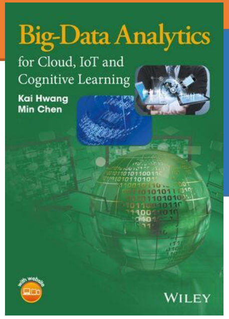
Big Data Requirements in Social-Media Applications

- Communication in social-media exchanges
  - Mobile social media communication takes two forms: business-to-consumer (B2C), in which a company may establish a connection to a consumer based on its location and provide reviews about user-generated content.
  - For example, McDonald's offered \$5 and \$10 gift cards to 100 users randomly selected among those checking in at one of its restaurants. This promotion increased sales by 33%, and resulted in many blog posts and news feeds through Twitter messages.



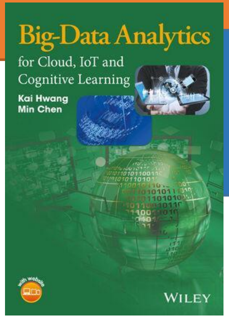
Big Data Requirements in Social-Media Applications

- Sales promotions and discounts
  - Although customers have had to use printed coupons in the past, mobile social media allows companies to tailor promotions to specific users at specific times.
  - For example, when launching its California-Cancun service, Virgin America offered passengers two flights to Mexico for the price of one.



Big Data Requirements in Social-Media Applications

- e-Commerce
  - Mobile social media applications such as Amazon.com and Pinterest have started to influence an upward trend in the popularity and accessibility of ecommerce, or online purchases.
  - Such e-commerce events could be conducted as B2B (business-to-business), B2C (business-to-customer), C2B (customer-to-business or C2C (customer-to-customer in a peer-to-peer (P2P) fashion). Recently, O2O transactions are also taking place as online to offline or offline to online sales or business exchanges.

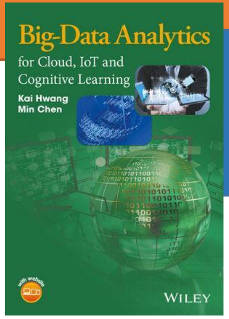


Big Data Requirements in Social-Media Applications

Table 8.3 provides a list of the leading 14 social networks based on the number of active user accounts, as of April 2016. Obviously, Facebook and WhatsApp are both very successful in attracting users. In China, the QQ and WeChat users are booming rapidly, which involves almost two-thirds of the population of China.

Table 8.3: Top 14 Social Networks based on Global User Population in 2016

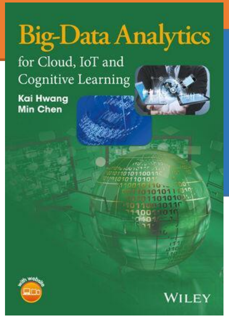
Social Network	**Active Users**	**Social Network**	**Active Users**
Facebook	1.59 Billion	**Twitter**	320 Million
WhatsApp	1.00 Billion	**Baidue Tieba**	300 Million
QQ	853 Million	**Skype**	300 Million
WeChat	697 Million	**Viber**	249 Million
Qzone	640 Million	**Sina Weibo**	222 Million
Tumblr	555 Million	**Line**	215 Million
Instagram	400 Million	**Snapchat**	200 Million



Social Networks and Graph Analysis

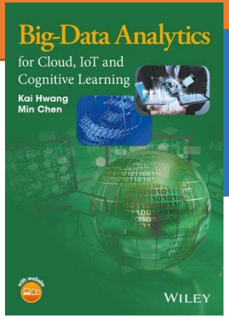
- The social network is useful in social sciences to study relationships between individuals, groups, organizations or even entire societies. The term is used to describe a social structure determined by such interactions. The ties between constituting members represent the convergence of various social contacts.

- An axiom to understanding social interaction is that it is based on the properties of relations between and within the social group.



Social Networks and Graph Analysis

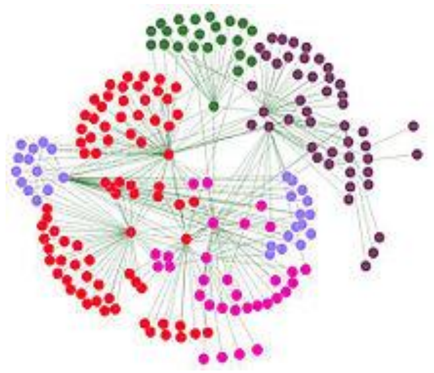
- Levels of Social-Media Networking
  - Social networks are analyzed at the scale relevant to the researcher’s theoretical question. Although levels of analysis are not necessarily mutually exclusive, there are three general levels into which networks may fall: micro-level, meso-level and macro-level.



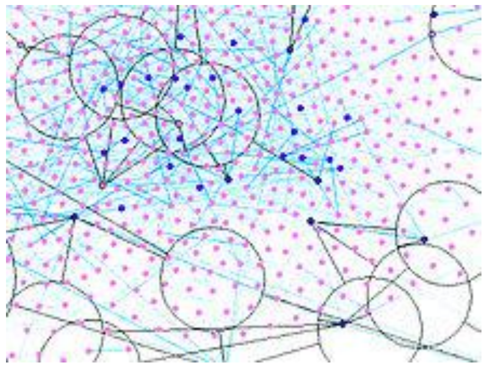
Social Networks and Graph Analysis

Example 8.8 Construction of Social Networks in Three Levels

At the micro-level, social network research typically begins with an individual, snowballing as social relationships are traced, or may begin with a small group of individuals in a particular social context.

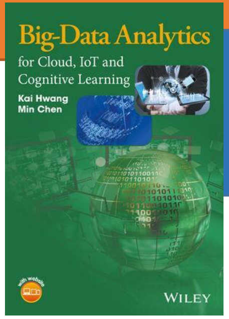


(a) Micro-level networks



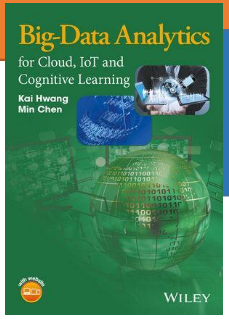
(b) Macro-level and meso-level networks

Figure 8.19: Micro-level, meso-level and macro-level construction of social networks



Social Networks and Graph Analysis

- Social Graph Characteristics
  - Social network analysis has emerged as a key technique in modern sociology. Characterizing the existing relationship among a person's social group is the main task in social network analysis.
  - The nodes in a social graph correspond to the users or actors and the graph edges or links refer to the ties or relationships among the nodes.



Social Networks and Graph Analysis

- Social Network Graph Properties
  - A social network is simply a map of all of relevant ties between all actor nodes. The network can also be used to measure social capital – the value that an individual gets from the social network. These concepts are often displayed in a social network graph.

    An example social-network graph is shown in Figure 8.20.

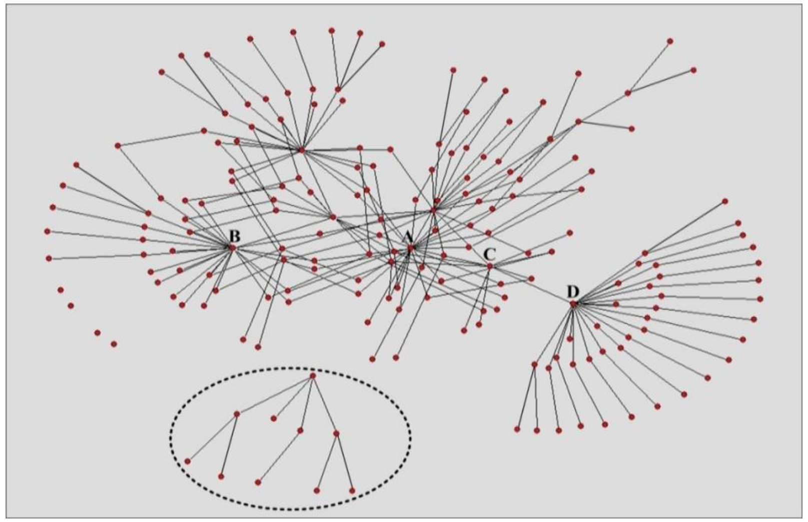
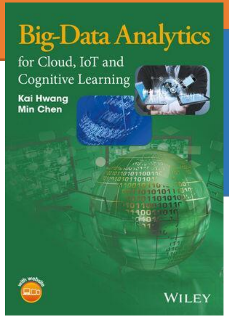
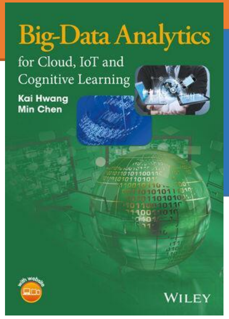


Figure 8.20 The graph representation of a social network



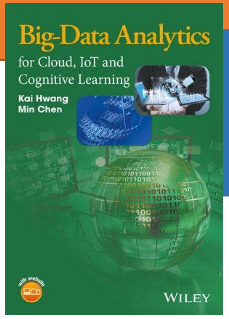
Social Networks and Graph Analysis

- Node Degree, Reach, Path Length and Betweenness
  - The node degree is the number of immediate node neighbors of a node. The reach is defined as the degree by which any member of a network can reach other members of the network.
  - Path length measures the distance between pairs of nodes in the network.
  - Average path length is the average of these distances between all pairs of nodes.
  - Betweenness reveals the extent to which a node lies between other nodes in the network.



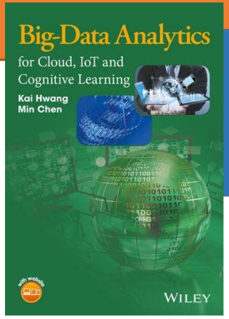
Social Networks and Graph Analysis

- Closeness and Cohesion
  - The degree an individual is near to all other individuals in a network (directly or indirectly). It reflects the ability to access information through the network members.
- Centrality and Centralization
  - Centrality indicates the social power of a node based on how well they "connect" the network. Nodes A, B and D in Figure 8.20 are all centrality nodes with different node degrees.



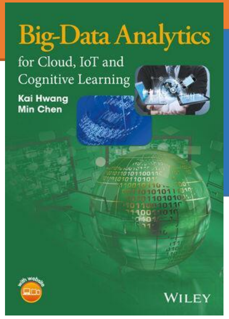
Social Networks and Graph Analysis

- Social Circles or clusters
  - This refers to some structured groups. If there is less stringency of direct contact or as structurally cohesive blocks, then a social circle can be created either loosely or tightly, depending on the stringency rules applied.
  - Those nodes inside the circle of Figure 8.20 form a cluster. The clustering coefficient is the likelihood that two associates of a node are associates themselves.
- Centralized versus Decentralized Networks
  - Centrality gives a rough indication of the social power of a node based on how well they "connect" the network. Betweenness, closeness and degree are all measures of centrality.



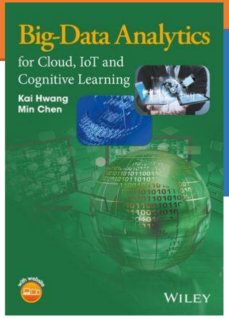
Social Networks and Graph Analysis

- Bridge and Local Bridge
  - An edge is a bridge if deleting it would cause its endpoints to lie in different clusters or components of a graph.
  - For example, the edge between nodes C and D in Figure 8.20 is a bridge. The endpoints of a local bridge share no common neighbors. A local bridge is contained in a cycle.
- Prestige and Radiality
  - In a social graph, prestige describes a node’s centrality. “Degree Prestige”, “Proximity Prestige” and “Status Prestige” are all measures of Prestige. Radiality is the degree to which a network reaches out and provides novel information and influence.



Social Networks and Graph Analysis

- Structural Cohesion, Equivalence and Holes
  - Structural cohesion is the minimum number of members who, if removed from a group, would disconnect the group. Structural equivalence refers to the extent to which the nodes have a common set of linkages to other nodes. These nodes do not have any ties to each other. A structural hole can be filled by connecting one or more links to reach other nodes.
  - This is related to the social capital: by linking two disconnected people, we can control their communication.



Social Networks and Graph Analysis

- Social Graph Analysis Example
  - Online social networking services are put together through identity, conversation, sharing, telepresence, relationship, affiliation, etc.
  - Listed below are some ideas in providing online social networking services:

1.Personal page or profiles for each user linked by social connections;

2.The social graph traversal along specific social links or networks,

3.Communication tools between the participants or registered users;

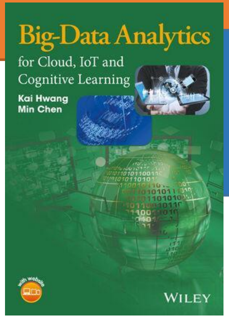
4.Share special information like music, photos, videos, etc. with friends or professional groups.

5.Operating a community in special niche topic areas such as healthcare, sports, hobbies, etc.

6.Customized software tools or databases may be needed to set up the social network services

7.Strong customer loyalty and viral membership growth are typical in social network communities.

8.Social networks have revenues by selling premium memberships and access to premium content.



Social Networks and Graph Analysis

Example 8.9: Graph Representations of An e-Mail Exchange Network

In Figure 8.21, we show a 430-node network of email exchanges within a small research laboratory. Only internal email exchanges are shown; emails to and from external sources are not shown here. The edges correspond to those who have sent emails to each other. The number of edges could easily reach 200,000 if fully connected. Here we see only a partially connected graph with roughly 5000 edges.

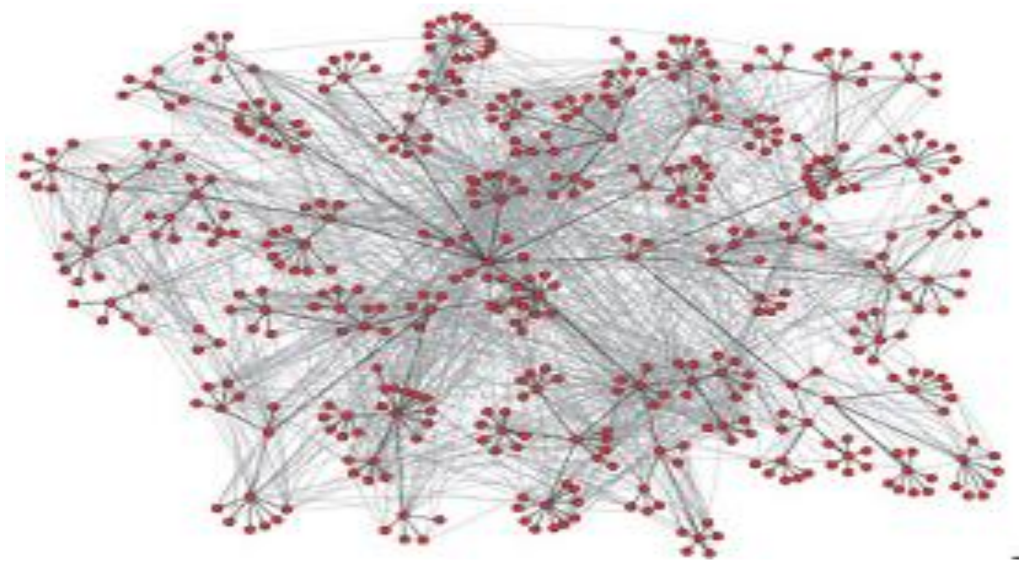
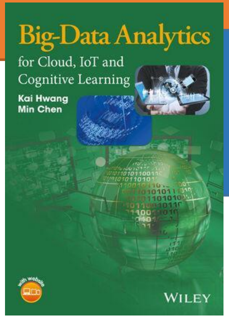
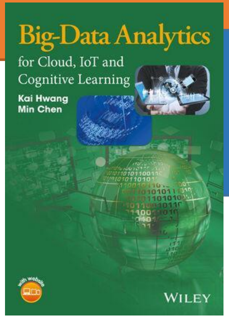


Figure 8.21 Graph representation of a social network



Social Networks and Graph Analysis

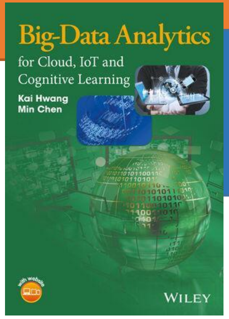
- Filtering Techniques and Recommender Systems
  - Social or collaborative filtering of unwanted data by polling the opinions of the masses to make decisions based on ratings. Content-based filtering is needed to recommend items based on features of products and ratings by users. Demographic filtering helps making decisions based on demographic information of the users. Finally, knowledge-based filtering makes wise decisions based on expertise knowledge and peer reputations, etc.
  - Hybrid filtering combines the advantages of the above filtering techniques to make even smarter decisions.



Social Networks and Graph Analysis

- Pushing Data Analytics for Cloud/Network Security Enforcement

  - This is a hot research area to apply big data for cyber security enforcement. Big data analytics is much needed in network security, enterprise events analytics, netflow monitoring to identify botnets, detection of persistent threats, data sharing, provenance, and governance techniques for trust management with reputation systems.

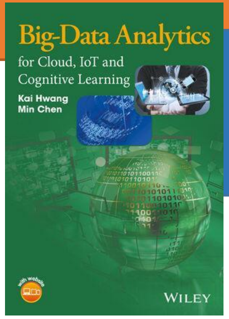


Social Networks and Graph Analysis

- Cloud Support of IoT and Social Network Applications

  - In cyber-physical systems (CPS), analytical algorithms can perform more accurately in system configuration, physical knowledge and working principles. To integrate, manage and analyze machinery, we desire to handle data more efficiently during different stages of the machine life cycle.

  - The coupling model between humans and machines is greatly facilitated by the use of cloud storage and analytics systems.This involves the sensing, storage, synchronization, synthesis and service operations.

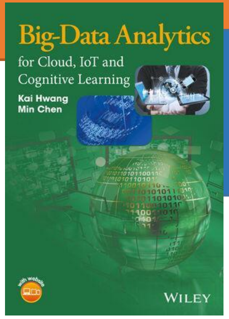


Social Networks and Graph Analysis

Online Social Network Architecture

- It is desirable to customize the OSN to sustain the competition in this field. The social network provider should choose a brand name with its own API interfaces and profile variables. The chosen forum categories must be relevant to the sufficiently large user community.

- The social network community must operate reliably with high availability and performance. Logically, the OSN provides a P2P platform. However, modern popular social networking services are all built with the client-server architecture for easy management and maintenance.

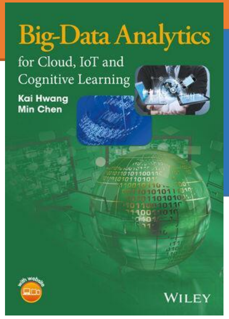


Social Networks and Graph Analysis

Example 8.10 Cloud Mashup for MapReduce Detection of Spam in a Message Stream

The idea of using MapReduce to filter out spam in a mashup of two cloud platforms is shown in Figure 8.22. Here, we assume unknown spam is embedded within a legitimate blog stream from Twitter applications. This data stream could be very large, in the TB or PB range, so we have to apply the Amazon EC2 cloud to implement a Naïve Bayesian classifier to detect the presence of spam in the data stream. The Baysian classifier is trained by comparing the detected results with some labeled samples to improve the classification accuracy.

On the input side at the left end, initial checking of URL links can eliminate some known spam. The blog stream flows through the MapReduce engine from left to right. We apply Map function to map out the input file to different machine instances, and the Reduce function for Naive Bayesian classifier construction.



Social Networks and Graph Analysis

The training of the Baysian classifier is done by using the feedback of detected results against some spam with known labels. With an input data streamof 1 TB, the EC2 spam detection can be done in less than 10 seconds, comparing 1000 seconds needed to detect the spam on a desktop computer. The detection accuracy achieved was reported to be higher than 90%.

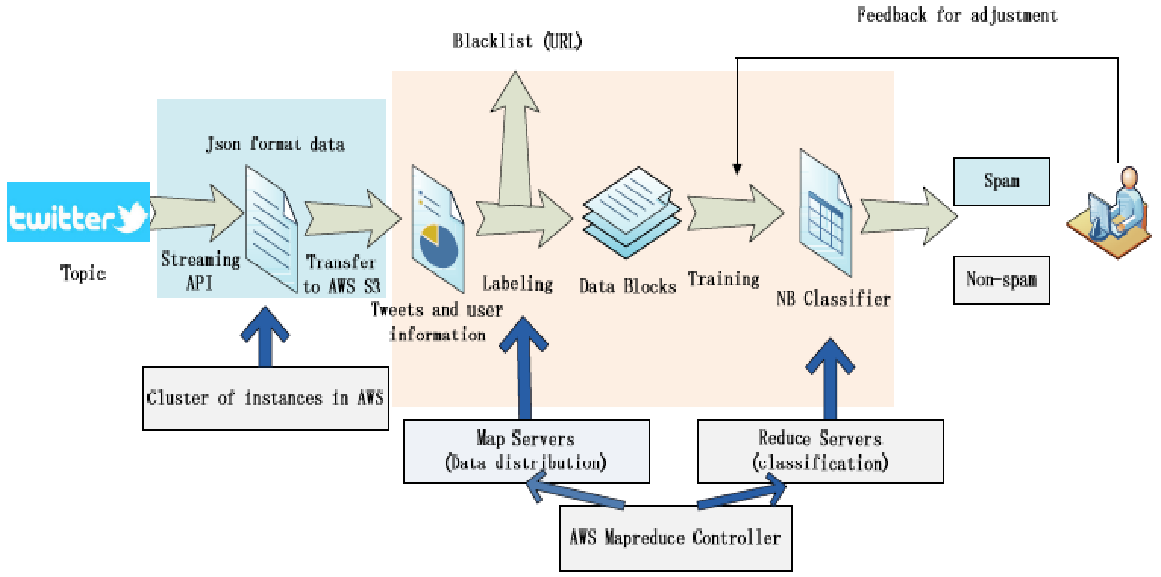
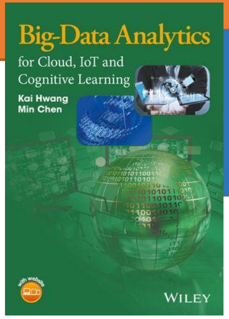


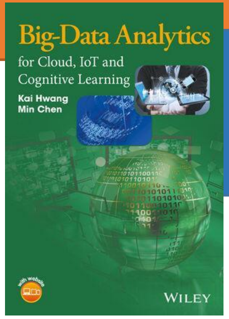
Figure 8.22: Detection of spam attacks from 1 TB of Twitter blogs on EC2 cluster in the AWS cloud using the Baysian clasifier. (Courtesy of Yue Shi and K. Hwang,IEEE Mobile Cloud Computing Conference, March 31, 2015)



Predictive Analytics Software Tools

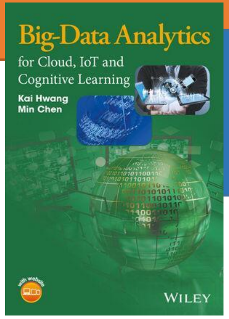
- Some commercial predictive analytics tools are introduced below. These tools are indispensable in social media and business applications of big data resources.

- Both open source and commercial analytics tools are available from large or small software companies or research organizations, such as IBM, SAP, Oracle,MATLAB, SAS, Predixion, etc.



Social Networks and Graph Analysis

- Predictive Analytics Applications
  - Important applications of predictive analytics software are listed below.
    - Analytical customer relationship management (CRM)
    - Clinical decision support and disease prediction
    - Fraud detection, loan approval, and collection analytics
    - Child protection, healthcare and elders caring
    - Customer retention and direct marketing
    - Portfolio, product or economical prediction
    - Underwriting and risk management

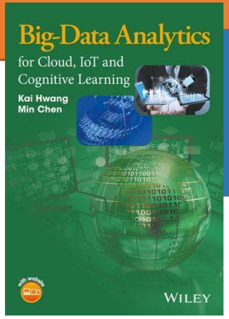


Social Networks and Graph Analysis

- Commercial Software for Predictive Analytics
  - In Table 8.4, we summarize the functionality and application domains of five representative predictive analytics software packages. These are selected from a long list of 31 predictive analytics packages reported at https://www.predictiveanalyticstoday.com/what-is-predictive-analytics/.

Table 8.4 Top 5 Commercial Predictive Analytics Software Systems .

Software Name	Functionality and Application Domains
IBM Predictive Analytics	Predictive analytics portfolio from IBM includes SPSS Modeler, Analytical Decision Management, Social Media Analytics, SPSS Data Collection, Statistics, Analytic Server, and Analytic Answers.
SAS Predictive Analytics	SAS supports predictive, descriptive modeling, data mining, text analytics, forecasting, optimization, simulation, and experimental design.
SAP Predictive Analytics	SAP Predictive predictive (lower-case p) analytics software works with existing data environment as well as with the SAP BusinessObjects BI Platform to mine and analyze the business data, anticipate business changes, drive smarter, and more strategic decision making.
GraphLab Create	A machine learning platform from Dato that enables data scientists and app developers to easily create intelligent apps at scale.
Predixion	The very cloud based predictive modeling platform back in 2010. It supports end-to-end predictive analytics capabilities from data shaping to deployment. Models evolved from machine learning libraries by Microsoft SQL Analysis Services, R and Apache Mahout.

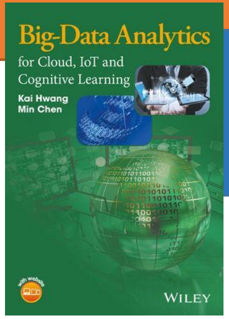


Social Networks and Graph Analysis

Example 8.11 SAS Analytics for Predictive and Descriptive Modeling

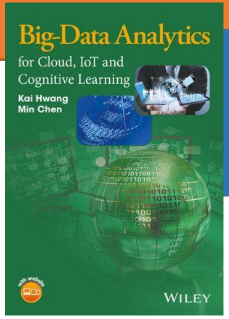
SAS Predictive Analytics provides a commercial software package for integrated predictive, descriptive modeling, data mining, text analytics, forecasting, optimization, simulation and experimental design.

The application domains of SAS Analytics include predictive analytics, data mining, visual analytics, forecasting, econometrics and time series analysis. The package can also be applied in model management and monitoring, operations research, quality improvement, statistics, text analytics and analytics for Microsoft Office.



Community Detection in Social Networks

- In social science, a community (or a cluster) is formed by a group of people under some bounding relations.The community structure is often represented by social graphs.
- In an autonomy sense, communities are subgraphs with higher cohesion with internal nodes and very light connection with the rest of the graph system.
- The formation of a community subgraph follows some similarity function among its nodes.



Community Detection in Social Networks

- As illustrated in Figure 8.23, six graph operations can change the graph topology. Like a human community, the social graph can also vary during its life cycle.

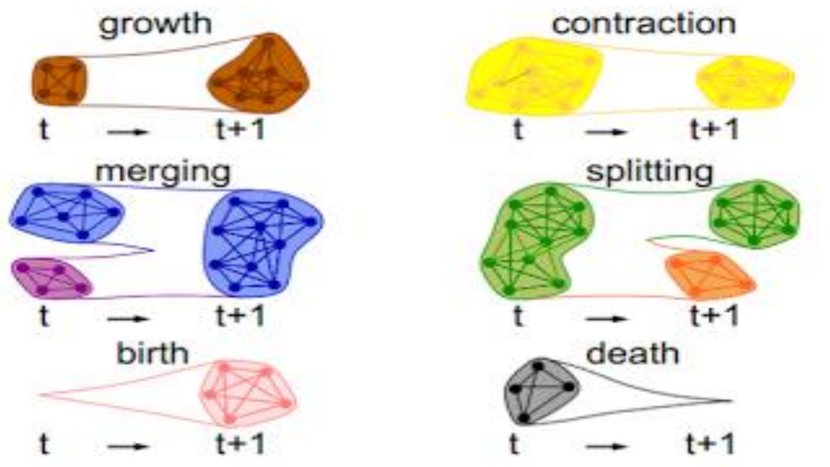
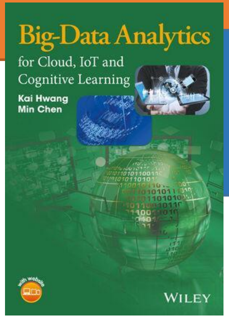


Figure 8.23 Forming a community graph by join, leave, growth, merging, splitting and contraction.

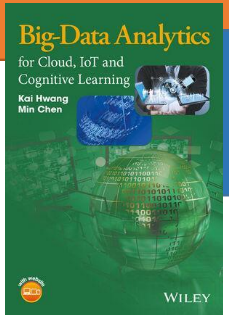


Community Detection in Social Networks

- Communities are defined as self-maintained subgraphs in a global graph. For social network analysis, we follow four subgraph properties: complete mutuality, reachability, node degree, and internal versus external cohesion in defining a community graph.

- Community detection refers to the process of detecting the existence of a community structure in a large social graph. A null model is used to verify whether the graph being studied displays a particular community structure or not.

- Modularity enables the detection of a community structure. Graph clustering is often performed based on modularity properties.

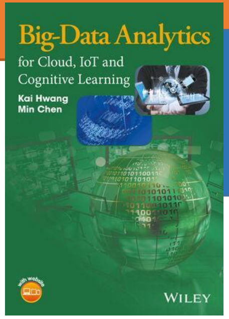


Community Detection in Social Networks

Example 8.12 High School Community Detection Based on Grade Classes

This example shows a simple social graph for grouping highschool children based on their grade classes. Each grade class is called a community here. The problem of community detection is to distinguish the grade classes based on the courses students are taking in the same year.

This is an overlapped community detection problem, because some students could be labeled as in 2 grade classes. The graph shown in Figure 8.24 partitions 69 students into 6 grade classes labeled Grade 7 to Grade 12.



Community Detection in Social Networks

Example 8.12 High School Community Detection Based on Grade Classes

The edges demonstrated their classroom relations by taking the same common courses. Obviously, students in the same grade are often taking a similar set of courses. Therefore, there exist more internal edge connections among them.

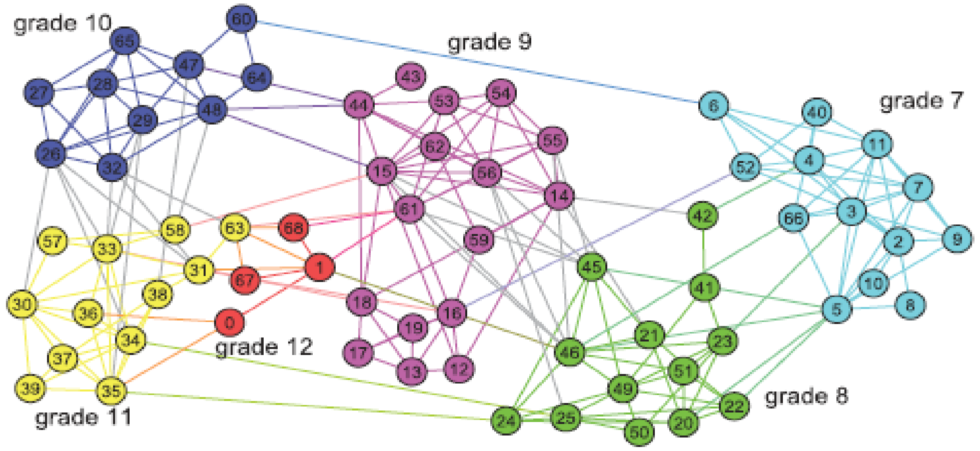
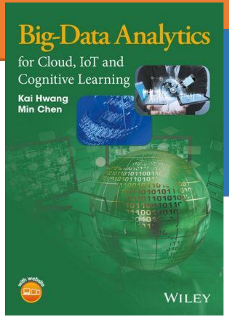
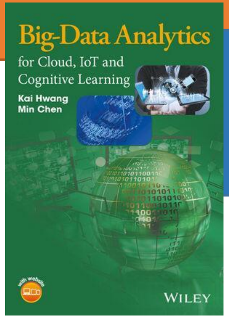


Figure 8.24 High school class formation based on the grade membership of students



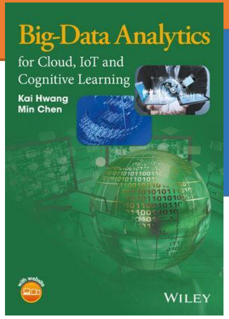
Community Detection in Social Networks

- To detect community affiliation in a social graph, we realize that non-overlapped communities are easier to detect than the overlapped cases, and list three methods to detect communities in a social graph.
  - Spin-spin model
  - Random walks
  - Synchronization



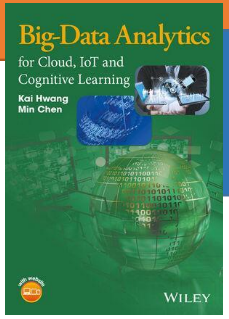
Community Detection in Social Networks

- Spin-spin model
  - A spinning system is used to spin among q possible states. The interaction is ferromagnetic, i.e. it favors spin alignment, so at zero temperature all spins are in the same state. If anti-ferromagnetic interactions are also present, the ground state of the system may not be the one where all spins are aligned, but a state where different spin values coexist, in homogeneous clusters.



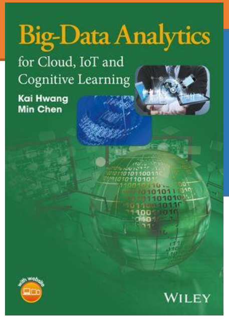
Community Detection in Social Networks

- Random walks
  - Random walks are useful to find communities. If a graph has a strong community structure, a random walker spends a long time inside a community due to the high density of internal edges and consequent number of paths that could be followed.
  - Here we describe the most popular clustering algorithms based on random walks. All of them can be trivially extended to the case of weighted graphs.



Community Detection in Social Networks

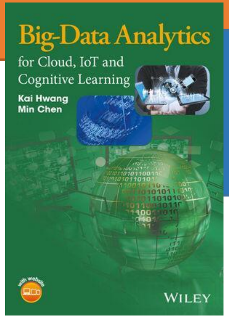
- Synchronization
  - In a synchronized state, the units of the system are in the same or similar state(s) at every time. Synchronization has also been applied to find communities in graphs.
  - If we follow the time evolution of the process, states with synchronized clusters of vertices can be stable and long-lived, so are easily recognized.



Community Detection in Social Networks

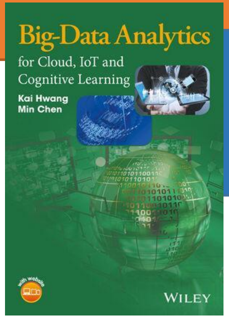
- The ultimate goal of clustering algorithms is trying to infer properties of and relationships between vertices, which are not available from direct observation/measurement. Still, there are also applications aimed at understanding real systems.

- Other social-media networks also exist in today's IT world and are briefly introduced below. These networks also generate big datasets that can feed into clouds in making analytics decisions:



Community Detection in Social Networks

- Collaboration Networks
  - In such a social network, individuals are linked together for common interest or business collaborations. Collaboration is done through an implicit objective concept of acquaintance.
- Citation networks
  - Citation networks have been used to understand the citation patterns of authors and to disclose relationships between disciplines.
- Legislative networks
  - enable us to deduce association between politicians through their parliamentary activity, which may be related or not to party affiliation.


Big-Data Analytics
for Cloud, IoT and Cognitive Learning
Kai Hwang
Min Chen
WILEY

Thank You !