HUMAN INTERACTION RECOGNITION BASED ON DEEP CONVOLUTION NEURAL NETWORK

Shen Shen

Dec. 17, 2017

Boston University

Department of Electrical and Computer Engineering

Technical Report No. ECE-2017-12

BOSTON
UNIVERSITY

HUMAN INTERACTION RECOGNITION BASED ON DEEP CONVOLUTION NEURAL NETWORK

Shen Shen

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
www.bu.edu/ece

Dec. 17, 2017

Technical Report No. ECE-2017-12

Summary

Semantic video based event recognition has drawn attention recently because of its spread application in solving real world problems. However, to solve this problem is challenging because the human behavior is complex and video data is very high dimensional data. To extract high level description from the video data, an effective way is to reduce the high dimension video data into feature in low dimension space. Convolution Neural Network is one way to automatically extract features into low dimension feature space. In this project, I present a way to apply 2D and 3D Convolution Neural Network to extract features from the video data and classify what type of interaction the video data belongs to. We test the model on SDHA data set that include 6 classes of different human interactions. The input of 2D CNN are gray-scale frames in the video segment and gray-scale frame stacks for 3D CNN. After then, the CNN classifying result of every frames or frame stacks in one video segment will be passed into voting process. This process will find the mode of the frame or frame stack prediction or use the maximum of the softmax probability summation as the voting result. The voting result is the classifying result of the video segment. The accuracy of 2D CNN for the 6 class classification is 65% and 3D cannot successfully classifying the video segment because of the limit of training sample number in the data set.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

My project of the EC720 course is to try to explore the dataset from the ICPR 2010 Contest on Semantic Description of Human Activities (SDHA) High-level Human Interaction Recognition Challenge. This challenge provides continuous videos captured by static camera on two scenario. Each video contains several human-human interactions (e.g. hand shaking) occurring sequentially and/or concurrently.[1]

The whole challenge requires contesters to correctly locate where the activity is happening and to annotate the occurring activity of each frame. Which means that the overall task of the data set could be separated into 3 part.

1. Where: figure out the spatial segment of the interaction
2. When: figure out the temporal segment of the interaction
3. What: classify what kind of action is it

This project will focus on the 'What' part of this challenge in the first scenario, so the task can be treated as a semantic video based event recognition problem. Semantic video based event recognition is to describe which type of event the video is, it has various real world application such as video surveillance and customer behavior analysis.

Deep neural networks has been proved very good at solving image classification problems[2]. It can automatically learn the parameter of the model by providing training data to extract spatial feature. Those feature can be used for reducing the object loss function of the training set and classifying the providing data. The idea of 2D CNN based video classification is to correctly classifying each frame in the video and use the mode on predict result or the summation of the prediction probability to find out which class the video is.

Also, people tried to expand the idea of 2D CNN on video analyzing by implementing a 3D kernel to extract not only spatial but also temporal feature from the video[3]. It requires providing a stacks of frames from the video as input and do maxpooling subsample also in 3D space. The final out put of 3D CNN model is same as 2D CNN. The final prediction can also been generated from frame stacks predict mode and probability summation.

This project attempt to implement both 2D and 3D CNN model to explore the semantic video based event recognition problem in SHDA data set.

Chapter 2

Dataset Description

2.1 Scene of the Video Dataset

10 videos with the resolution of 720*480, 30fps are provided by the data set. It contains 6 classes of human-human interactions: shake-hands, point, hug, push, kick and punch. The ground truth labels for each interactions, the coordinate of where each interaction is happening and the start and end time of each interaction is also provided. Each video contains 6 executions on average. In total, there are 60 executions, 10 for each class.

Because the coordinate of the interaction and the start and end time of every interaction is provided, we can use those spatio-temporal boundary to crop small segments that contain exactly one interaction execution. In the whole report, the 10 video are expressed as video sequences and those 60 segments of execution cropped from the video sequence are expressed as video segments. Figure 3.4 shows the example of video sequence and segment. Each video sequences lengths are around 1 minute. Each video segment lengths are around 2 second.

(a) (b)

Figure 2.1: (a)frame in video segment; (b) frame in video sequence;

The 10 video sequences are taken at same scene on a parking lot. Generally the 10 video sequences are static but they involve slight camera movement.

2.2 Interaction

Figure 2.2 shows all the 6 types of interactions in the video sequences and their spatial boundary. Among all 6 activities, only Pointing could be finished by only one person. The ground truth bounding box of Pointing has a very different height weight ratio with other activity.

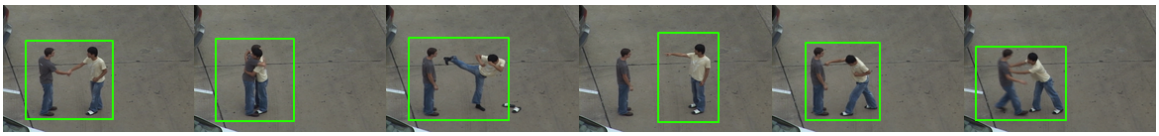


Figure 2.2: example of all six interactions

Chapter 3

Deep Neural Network

3.1 Artificial Neural network

Artificial neural networks is a networks system that combined with many small unit called neuron. It could be use for supervised classification problem. A simple example Artificial neural network for classification is shown as Figure 3.1. This network called multi layer perceptron(MLP) or fully connected network. The cell in each layer is called neuron.

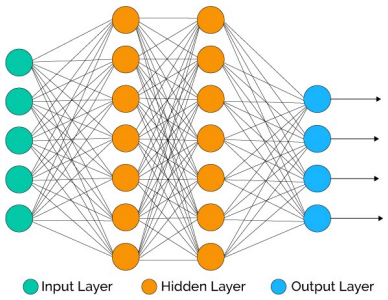


Figure 3.1: example of fully connected network

3.1.1 Feed Forward

Assume the input feature $X \in R^d$, and the output $y \in 1, 2...m$ indicating which class is the sample, The fully-connected network could be written as a function form $f : X \to Y$. When the input data pass through layers of an Artificial Neural Network, it pass through each hidden layer consist of lineal combination and activating function and the last layer is a lineal combination of the previous layer. Each neuron apply those calculation to the input of its layer. After passed through all the layer, the output should has the same shape as the one hot category of the label.

$$h_1(X) = activation(W_1 X + b_1) \tag{3.1}$$

$$h_2(X) = activation(W_2(h_1(X)) + B_2) \tag{3.2}$$

$$h_3(X) = W_3(h_2(X)) + B_3 \tag{3.3}$$

The activation function is an nonlinear function, one wildly use activation function is Relu function shown as Figure 3.2

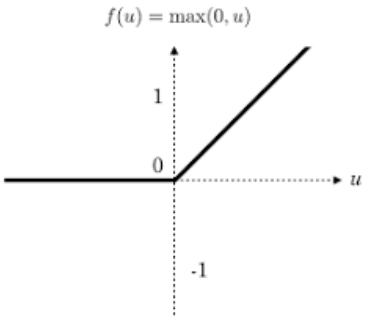


Figure 3.2: Relu function

Then, the softmax of the output would be the predict probability of the input data belongs to the class on that index and the max probability index show which class the data belongs to.

3.1.2 Back Propagation

To learn the parameter in the ANN model, the model can be train by backpropagation algorithm. This algorithm calculates the gradient of the Loss function to each parameter by using chain rule. In classification problem, the cross entropy loss Equ 3.6 is wildly used.

$$Loss(p, y) = -(y \log(p) + (1 - y) \log(1 - p)) \tag{3.4}$$

By taking chain rule, we can modify the parameter in gradient decent way:

$$W_{t+1} = W_t + \eta \frac{\partial Loss}{\partial W_t} + \xi \tag{3.5}$$

W could be every parameter that need to be trained and ξ is a small stochastic term.

3.1.3 Regulization

One sever problem of ANN is the over fitting problem, if the neuron at last layer is too much, the last layer would just try to remember the result instead of extract effective feature. One way to solve this issue is to induce the complexity as a penalty term in loss function. then the Loss function become:

$$Loss(p, y) = -(y \log(p) + (1 - y) \log(1 - p)) + reg \tag{3.6}$$

The regularization term could be L1 or L2 norm of all the parameter.

The other way to avoid overfitting is to add drop out layer. This layer will block the output of some neuron randomly when training. It prevent the neuron to memory the output and improves the generalization ability of the model.

3.2 2D Convolution Neural network

Although fully connected network is effective in many classification task. Problems arise when we apply fully connected network to image and video data classification problem. One reason is that if we want to do lineal combination calculation on high dimension data like image and video, the parameter should also be a very huge matrix. More computational intensity and memory is required.

The idea of Convolution Neural Network(CNN) is that instead of doing full matrix product, this model apply convolution calculation with convolution kernel and use maxpooling to capture the feature. It reduce the number of parameter from related to the input size of each layer to the given kernel size.

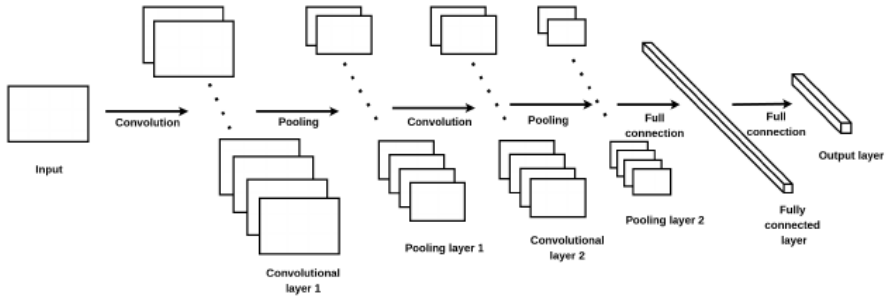


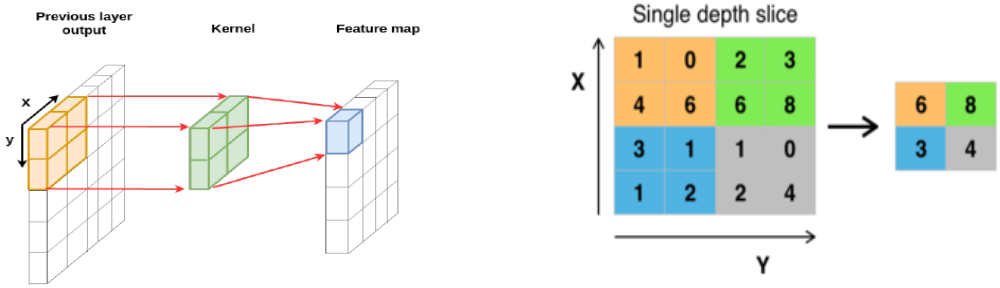
Figure 3.3: Convolution Neural Network

The convolution layer in 2D CNN is shown as Equ 3.7

$$y = f(\sum_i \sum_j w_i j v_{x_1+i, x_2+j} + b) \tag{3.7}$$

After pass the data through the convolution layer, we will do maxpooling on the data, max pooling is to find the maximum value in a sliding pooling window. This could find the most significant convolution feature in a window. By taking maxpooling with stride, we can reduce the dimension of the original data,

The parameter of the CNN network will also learned by taking back propagation. The dimension of the original data reduce into many convolution feature after pass through all the layer. Then, the we can flatten all those feature as input of a fully connected layer for classification task. This model generate the kernel that could distinguish the image in order to effectively classify the image.



(a) (b)

Figure 3.4: layer in CNN: (a) convolution layer; (b) maxpooling layer;

3.3 3D Convolution Neural network

3D convolution is an extension to the 2D convolution. It uses 3D kernel to capture not only spatial but also temporal feature. The back propagation will modify all the parameter of the 3D convolution kernel and bias. The input of the model should be modified as a stack of consecutive frames. The maxpooling layer of 3D CNN model should be a 3D maxpooling. Fig 4.3 show how a 3D concolution kernel works on the input data.

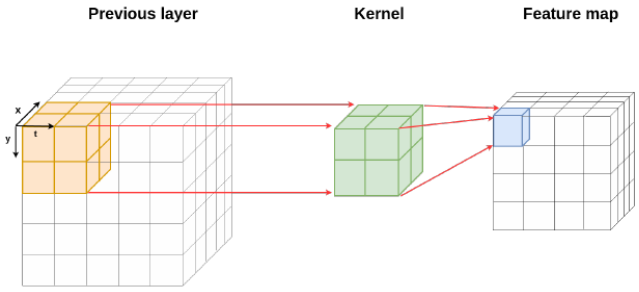


Figure 3.5: 3D convolution layer

Chapter 4

Implementation

By implementing the model to the SDHA dataset, we need to generate training and testing samples. In order to evaluate the model, we apply 10-fold leave-one-out cross validation on the given data set.

Gray scale image and image stacks is enough for the classification task. Because every video segment has a different shape, we have to reshape all the frames or frames stacks into same size.

Every video segment has different length from 65 frame to 185 frame, on average it is 121 frame.

In order to train a CNN, the original data is not enough, so a data augmentation is required. An augmented training set is generated by randomly crop, scale, shear and rotate the images in the training set. One image could be used to generate 20-40 augmented image. The augmentation number of each class is decided with the aim to make training sample number in each group close.

4.1 2D Convolution Neural Network

The training of the 2D CNN use Nvidia DIGITS Deep learning training system with caffe model, I design a convolution neural network with 2 convolution layer. Overall, the model structure of the 2D CNN is shown as Fig 4.1. Each conv cell indicate the convolution kernel size, Pool2 stand for 3*3 max pooling with stride 2.

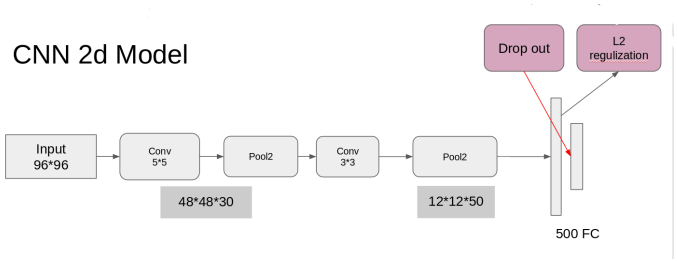


Figure 4.1: 2D CNN structure

After the model is defined, I try Implementing the model on the data set but I only get a very poor result: without any regularization, the training loss of both 2D and 3D CNN increases but the output on the test set is not better than randomly guessing. However, once doing regularization like drop out, The training loss can never decrease and the training accuracy remain at 17%. it shows that the model again doing randomly guessing.

The reason is that although one video segment shows exactly one interaction, the frames or frame stacks at the very beginning and end of the video segment do not show any useful information to indicate which type of interaction this video segment belongs to. Every time those frames or frame stacks pass through the CNN, they might receive a high loss and the back propagation process would tune the parameter to a bad value.

To solve this problem, we have to separate the irrelevant frame or frame stacks from useful one. The augmented sample number is around 10,000 per class.

Over all, the implementation of 2d CNN is:

1. Decode video to grayscale images, separate into training set and testing set(9 video to train, 1 video to test, leave one out).
2. Label all the images into 7 class, prepare one class for irrelevant frame
3. Randomly crop, scale, shear and rotate the images in training set to generate more training sample
4. Split 15% of the training set as validation set
5. Train the 2d CNN model, choose the model performs best on validation set
6. Test the decoded picture of the test video segments, save the softmax result
7. Sum up the softmax result or use the mode of prediction on each frame to predict which class is the video segment belongs to

4.2 3D Convolution Neural Network

To implement 3D CNN, we need to change the input as frame stacks, the input frame stacks should be selected from the original video segment. 8 frame stacks is adequate to recognize what kind of interaction the stack is. In order to prevent the input data stacks to be too similar, a sliding window with stride could be used. This method would first take frame 1 to 8 into a frame stack, then take a stride 4 and combine frame 5 to 12 into a stack an so on, Fig 4.2 is the sketch of how the sliding sample window works for the video segment.

I also try to implement throw out irrelevant on the 3D CNN model. However, since we have to use sliding window in 3D CNN to generate video stacks, the sample

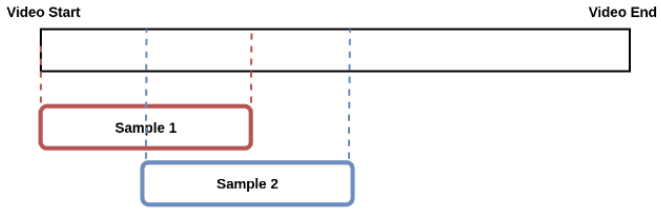


Figure 4.2: Sample window for frame stacks

would be insufficient even when data augmentation is applied. So I only apply this effort on 2D CNN model.

The training of the 3D CNN use tensorflow API, is not complicated scene, 3D CNN also down sample on temporal space. In order to down sample the 8 frame stack into one, I implement 3 convolution layer. Overall, the model structure of the 3D CNN is shown as Fig 4.3. Pool 2 2 1 stand for pooling with stride 2 on 2D space, Pool 2 2 2 stand for pooling on 3D space. This model refers to [5].

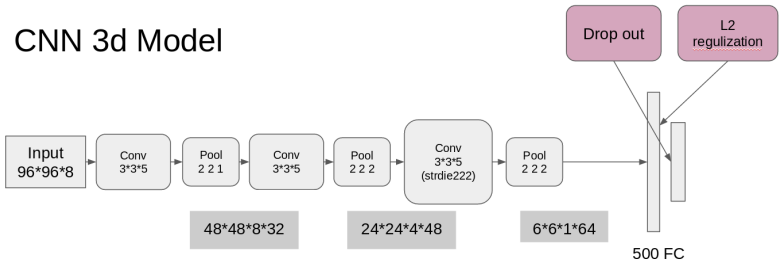


Figure 4.3: 3D CNN structure

Over all, the implementation of 3D CNN is:

1. Extract frame stacks from the training video sequence. Label the frame stacks.(leave one out)
2. Label all the images
3. Randomly crop, scale, shear and rotate the images in training set to generate more training sample
4. Split 15% of the training set as validation set
5. Train the 3d CNN model, choose the model perform best on validation set
6. Extract sample from the testing video segments, pass them through the 3D CNN model, save the softmax result.
7. Sum up the softmax result or use the mode of prediction on each frame to predict which class is the video segment belongs to

Chapter 5

Result

5.1 2D CNN

The training process of 2D CNN is shown as Fig 5.1. blue line for training loss, green line for validation loss and orange line stand for validation accuracy.

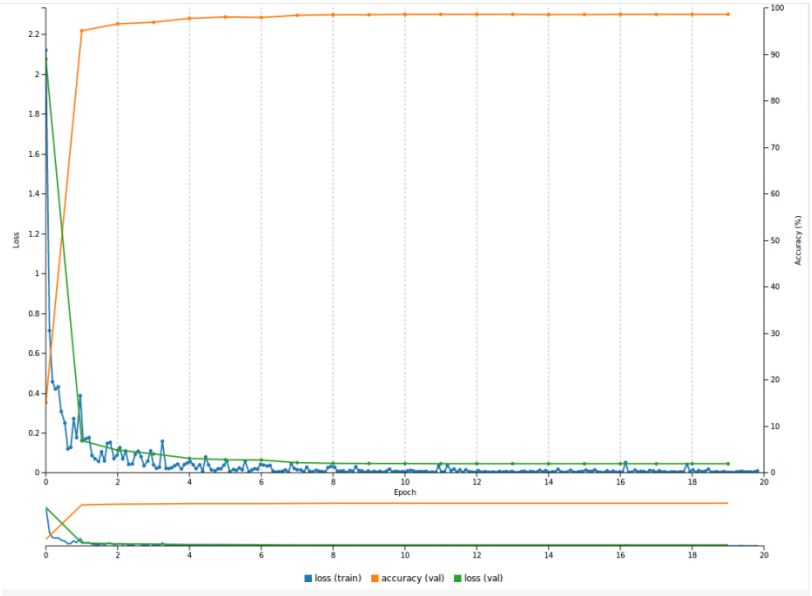


Figure 5.1: The training result of 2D CNN

As the training processing, the validation accuracy stop at near 98%. By training 10 model on 10 leave one out testing set. we get a confusion matrix as Table 5.1. The result of the confusion matrix is the mode of the prediction result. Table 5.2 reveal the accuracy on each class. The result is completely no different between prediction voting and max probability summation on every single video segments.

Table 5.1: Confusion matrix of 2D CNN

	handshake	hug	kick	point	punch	push
handshake	5	0	4	0	0	1
hug	0	7	2	0	0	1
kick	0	0	8	0	0	2
point	0	0	0	10	0	0
punch	0	1	0	0	4	5
push	0	1	2	0	2	5

Table 5.2: test accuracy of 2D CNN

	handshake	hug	kick	point	punch	push	average
perdiction mode acc	0.5	0.7	0.8	1	0.4	0.5	0.65
max prob acc	0.5	0.7	0.8	1	0.4	0.5	0.65

5.2 3D CNN

The training on 3D CNN fail. But the loss function and the training performance could show us how a overfitting model looks like. as Figure 5.2 shown, the loss would not be smooth for a overfitting model. but for some training sample, the loss is near completely zeros.

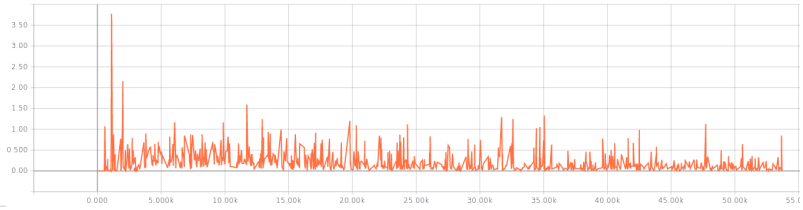


Figure 5.2: 3D CNN loss function

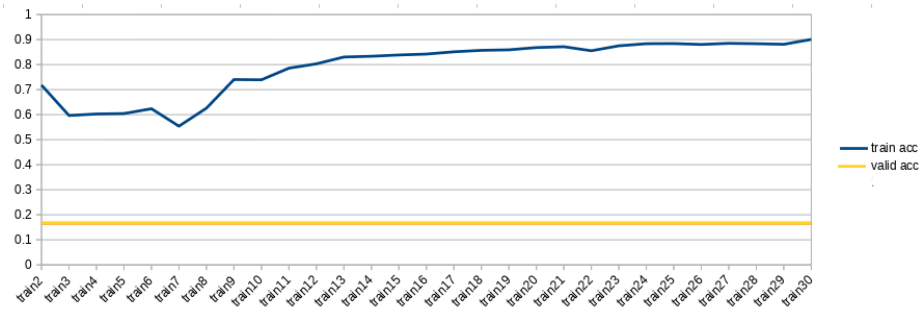


Figure 5.3: 3D CNN training and validation accuracy

Chapter 6

Conclusion

By using 2D and 3D CNN on human interaction classification problem, I learned how to build and train a deep neural network model and use the model to predict result.

The project shows how 2D and 3D CNN can extract features from the video data and do classification by providing high dimension image and video data. To build a CNN model, a large amount of training samples should be provided so data augmentation is needed. Irrelevant samples impact the performance of CNN.

We train 2D CNN by picking out the irrelevant frame as a new group, and augmenting the training sample to around 10,000 sample per class. After training, the testing accuracy on the SDHA data set on our 2D CNN frame prediction voting model is 65%, while hough voting model from [6] achieve 89% accuracy.

The 3D CNN model failed to converge because of the sample number limitation after throwing out irrelevant training samples .

Forward work could be a try to apply 3D CNN on dataset with training sample. Moreover, instead of gray-scale image data sample, the feature related to motion like optical flow could also be treated as input for the CNN model.

Bibliography

[1] Ryoo, M. S. and Aggarwal, J. K., UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA), 2010

[2] Krizhevsky, Alex. "ImageNet Classification with Deep Convolutional Neural Networks. Retrieved 17 November 2013.

[3] Ji, Shuiwang; Xu, Wei; Yang, Ming; Yu, Kai, 3D Convolutional Neural Networks for Human Action Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 35 (1): 221231.

[4] Zhao Rui, Ali, Haider. Two-Stream RNN/CNN for Action Recognition in 3D Videos. arXiv:1703.09783, 03/2017

[5] Ryoo, M. S. and Aggarwal, J. K. ,"Spatio-Temporal Relationship Match: Video Structure Comparison for Recognition of Complex Human Activities", IEEE International Conference on Computer Vision (ICCV),2009, Kyoto, Japan

[6] Waltisberg D., Yao A., Gall J., Van Gool L. Variations of a Hough-Voting Action Recognition System. In: nay D., ataltepe Z., Aksoy S. (eds) Recognizing Patterns in Signals, Speech, Images and Videos. Lecture Notes in Computer Science, vol 6388. Springer, Berlin, Heidelberg (2010)