GALAXY CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS

BY

HONGHUI SHI

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Adviser:

Professor Thomas S. Huang

ABSTRACT

Galaxy classification, using digital images captured from sky surveys to determine the galaxy morphological classes, is of great interest to astronomy researchers. Conventional methods rely heavily on a few handcrafted morphological features while popular feature extraction methods that developed for natural images are not suitable for galaxy images. Deep convolutional neural networks (CNNs) are able to learn powerful features from images by hierarchical convolutional and pooling operations. This work applies state-of-the-art deep CNN technologies to galaxy classification for both a regression task and multi-class classification tasks. We also implement and compare the performance with several different conventional machine learning algorithms for a classification sub-task. Our experiments show that convolutional neural networks are able to learn representative features automatically and achieve high performance, surpassing both human recognition and other machine learning methods.

To my family, especially my wife, and my friends near or far.
To my adviser, to whom I owe much thanks!

ACKNOWLEDGMENTS

I would like to acknowledge my adviser Professor Thomas Huang, who has given me lots of guidance, support, and visionary insights. I would also like to acknowledge Professor Robert Brunner who led me to the topic and granted me lots of help during the research. Lastly I want to thank my colleagues and friends Wei Han, Thomas Le Paine, Pooya Khorrami, Xianming Liu, Yingzhen Yang, Shiyu Chang, and Yang Zhang. Without their friendship and support, this thesis would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
SDSS	Sloan Digital Sky Survey
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine

CHAPTER 1

INTRODUCTION

Image classification is a significant and recurring theme in pattern recognition and digital image processing, with many applications in a number of domains related to images, including computer vision, medical image analysis, biology, etc. In the field of astronomy, galaxy classification, using digital images collected from large-scale sky surveys to determine the galaxy morphological classes, has long been of great interest to astronomy researchers.

There are more than 170 billion galaxies in the observable universe, and the astronomical community has captured image data covering more than a quarter of the whole sky with powerful telescopes and ambitious sky surveys such as the Sloan Digital Sky Survey (SDSS) [1]. To tackle the galaxy classification task, astronomy experts Fukugita, Nair, Baillard, etc., independently classified thousands of galaxy images by themselves [2],[3],[4], and the Galaxy Zoo project used crowdsourcing to collect more than 60 million classification results from online citizen scientists [5],[6]. On the other hand, researchers have been investigating automated classification methods in the past two decades using popular machine learning algorithms along with dozens of handcrafted morphological features [7],[8],[9], and the results have contributed to the study of astronomy [5].

However, conventional automated classification algorithms are inadequate in terms of classification accuracy because, to a certain extent, they rely heavily on handcrafted features: feature extraction is hard for galaxy images with many visual subtleties such as large intra-class variance, similar appearance, and gradual change between different galaxy classes. Convolutional neural networks (CNNs)[10], along with the recently developed high performance GPU implementations [11], have enabled us to implement deep network architectures to learn representative features from images automatically and achieve high accuracy in image classification.

In this work, we implement deep CNNs on galaxy datasets for regression

and multi-class classification tasks. We also compare the performance of several different machine learning algorithms for a binary galaxy classification task. The experimental results demonstrate the feature extraction power of CNNs in terms of their learned feature filters and classification accuracy for galaxy images.

The remainder of the thesis will be organized as follows. Chapter 2 discusses the datasets for galaxy classification. Chapter 3 studies different classification algorithms with a focus on algorithmic performance aspects and then discusses CNNs in detail. In Chapter 4 and Chapter 5, we present implementation details and results for galaxy classification as regression and multi-class classification tasks. Finally, we conclude the thesis in Chapter 6.

CHAPTER 2

GALAXY CLASSIFICATION DATA

The Sloan Digital Sky Survey (SDSS) mapped nearly one quarter of the entire sky and included millions of images in five different bands in its data release 7 (SDSS DR7) [1]. The survey contains a huge amount of information as well as image data (more than 40TBs of galaxy images and catalogs, etc.), which is difficult to process in bulk and to analyze directly for scientific research.

Galaxy Zoo Project 2 is a recent citizen science project hosted by Galaxy-Zoo [12] to classify around $200,000$ RGB galaxy images from both SDSS DR7 and other similar surveys. The project collected more than 60 million human classifications within 14 months, and published its results in 2013 [12]. In December 2013, Kaggle [13] started a worldwide challenge for the automated classification algorithms using almost half of the data from the Galaxy Zoo Project 2 [13].

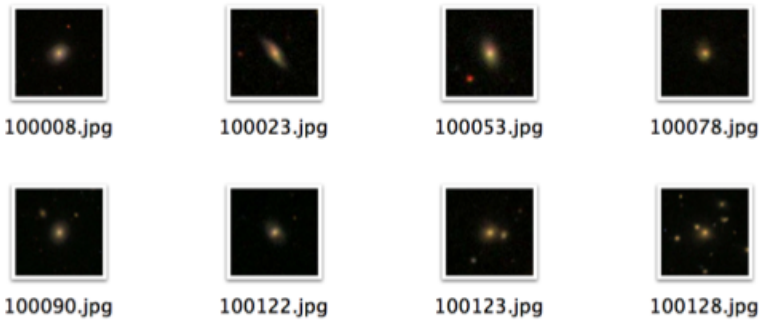

Figure 2.1: Galaxy image samples from Kaggle dataset

In this challenge, the training dataset consists of $61,578$ JPEG images of size 424×424 with RGB channels; each image has an ID number and a galaxy (or like shape) in the center (see Figure 2.1), in addition to 37 probability labels from volunteer classification results. The probability label for each image is generated by averaging the votes from about 44 volunteers who

answer 11 questions regarding classifying the corresponding image, and by post-processing techniques over the whole probability distribution [5]. The tree structure of the 37 probabilities and 11 questions is demonstrated in Figure 2.2.

The test dataset in this challenge consists of $79,975$ images from the same Galaxy Zoo Project 2 but without giving the probability labels. The task of this challenge is to predict all $79,975 \times 37$ probability labels for the test dataset, and the result is evaluated by calculating the root mean squared error (RMSE) over all predictions.

Besides experimenting with regression over all the given labels and images in the training set from the challenge dataset, we generate subsets to perform the multi-class classifications.

For the 2-class classification subtask, we first choose the two most important galaxy classes among the 37 classes to study: the "smooth galaxy" class and "galaxy with a feature or disk" class. These two classes are related to early-type and late-type galaxies, respectively, which are of greatest research interest to astronomers. We then generate our ground truth categorical class label from the given probability labels. More specifically, in accordance with some astronomical research conventions [9], we choose the subset of images and labels with label probabilities that are greater than 0.8 for the chosen classes. Finally, we get our new dataset with $24,273$ galaxy images and its corresponding labels for the chosen two classes. We perform classification tasks based on the generated new datasets with classification accuracy as the performance measure.

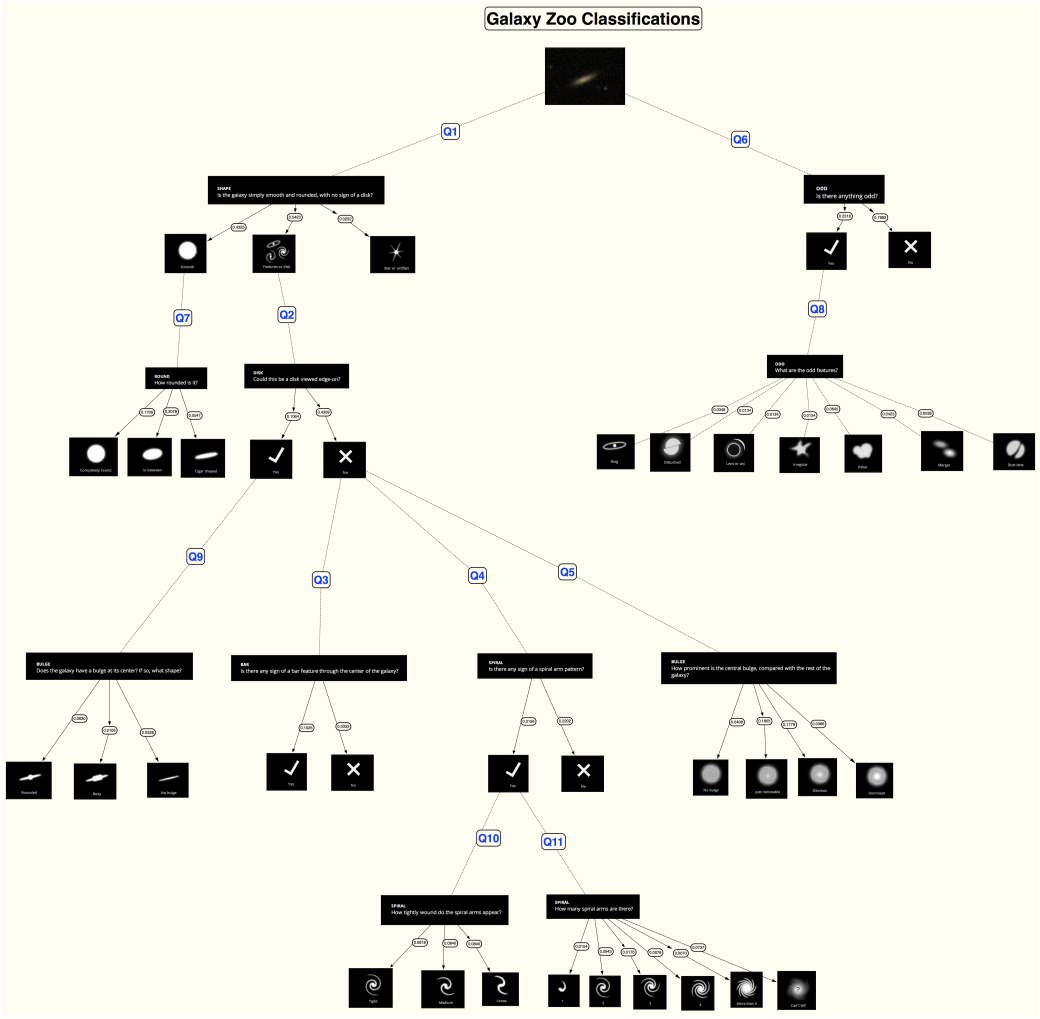

Figure 2.2: The ontology of galaxy classes. The nodes represent the 11 questions and the 37 answers derived from the Galaxy Zoo dataset (better viewed when zoomed in).

CHAPTER 3

CLASSIFICATION METHODS

In this chapter, we will briefly introduce the conventional learning algorithms we used in the classification task with a focus on their algorithmic performance. We will also discuss CNNs in detail.

3.1 Conventional Classification Algorithms

3.1.1 Nearest Neighbor

Nearest neighbor (NN) is one of the simplest classification algorithms. NN is non-parametric and needs no training. It predicts the label of a given observation to be the class label of the closest neighboring observation from the training set in the feature space. From the rules it is obvious that NN assumes nearest proximity of samples with the same label. The computational complexity of the NN algorithm depends on the underlying search algorithm we are using.

3.1.2 Logistic Regression

Logistic regression is a probabilistic linear classifier which projects the input vector onto a set of hyperplanes. Each hyperplane corresponds to a class, and the distance between the input vector and the hyperplane corresponds to the probability of the input belonging to that class. The complexity of solving the linear regression as an optimization problem depends on the optimizer; for example, using the L-BFGS we can get linear time on the size of the training set [14].

3.1.3 Naive Bayes

Naive Bayes is a family of probabilistic classifiers based on the strong naive Bayesian assumption that features of each dimension are conditionally independent given the label. The complexity of training is $O(n)$.

3.1.4 Decision Trees and Random Forest

Decision tree is an intuitive and popular method for classification. It maps the training dataset to a decision-making tree structure where the leaves represent the class labels, and the branches represent the conjunction of different features. Many versions of decision tree algorithms are available; for the C4.5 algorithm the complexity is $O(n \times d^2)$ [15], where n is training set size and d is number of features. Random forest [16] is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes of trees during prediction.

3.1.5 Support Vector Machines

Support vector machines (SVMs), along with the "kernel trick," are one of the most powerful classification algorithms introduced by Vapnik [17]. For the linear case, SVMs maximize the margin between the feature vectors in the feature space. And for the non-linear case, SVMs, by using the kernel trick, implicitly map the inputs into a higher dimensional space. Many optimization implementations are available for SVM, for example, $O(n^2 \times d)$ when using the RBF kernel with SMO solver, and $O(n \times d)$ for linear SVMs [18].

3.2 Convolutional Neural Networks

Inspired by the hierarchical human visual recognition system, convolutional neural networks are naturally designed for images. The idea of CNNs is to build models with multiple and hierarchical levels of abstraction of data representation from input images, to learn representative and adaptive features from data automatically and hence to improve the performance of later

classification action in the model.

Compared with traditional fully connected feed-forward neural networks, a CNN mainly introduces two types of layers with restricted connectivity: the convolutional layers and pooling layers.

3.2.1 Local Connectivity

A convolutional layer introduces local connectivity and sharing parameters [10], where local connectivity (each point in feature maps only corresponds to a local patch of the original image) enables us to discover and represent the 2-D topology structure, and sharing parameters (the use of the same filter for each feature map) reduce both the model parameter size, enabling more efficient computation, and the overfitting issue that might arise due to too many parameters and insufficient data.

As we can see in equation 3.1, Z^l is the activations in layer l, h is the activation function, and W^l is the shared convolutional filters for layer l.

$$Z^{l+1} = h(Z^l * W^l) \tag{3.1}$$

3.2.2 Hierarchical Structure

As illustrated in Figure 3.1, the first convolutional layer in the network corresponds to low level and local features, but in later convolutional layers, it will hierarchically represent more high level concepts and more global features as the convolution operations continue.

Then we use pooling layers to reduce the spatial resolution of an input feature map; thus, we can do the dense convolution operation first to find useful features and concepts and then reduce the model capacity by pooling to prevent too many parameters. Pooling also introduces certain spatial invariance to features positions. Pooling operations are special types of convolutional operations with fixed weights for the fitlers, such as max pooling and average pooling.

Convolutional layers and pooling layers are essential components of the CNN architecture; different choices of their numbers, sizes, and orders lead to different efficiency and accuracy of the final performance.

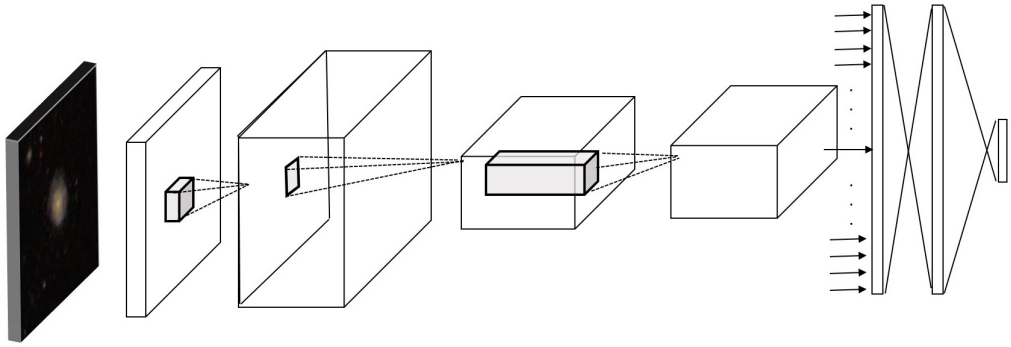

Figure 3.1: The hierarchical architecture of CNNs

3.2.3 Nonlinear Activation Units

Another significant feature of CNNs is the rectified linear unit (ReLU) activation function specified in equation 3.2. ReLU can simplify computation, introduce sparsity in the networks and, unlike the sigmoid and other functions, does not suffer from gradient vanishing.

$$h(a) = \max(0, a) \tag{3.2}$$

3.2.4 Training Algorithm

To train the CNNs, we use mini-batch stochastic gradient descent (SGD) with the back propagation algorithm [19] to calculate the gradient. SGD is the stochastic version of the general gradient descent using mini-batches of the training dataset instead of using all of it. The gradient descent updating formula is stated in equation 3.3, where W^t is the parameters at iteration t, and α is the learning rate which we anneal using a certain strategy.

$$W^{t+1} = W^t + \alpha \nabla W^t \tag{3.3}$$

We can further apply other techniques like data augmenting and cross validation to address the overfitting issues.

CHAPTER 4

GALAXY CLASSIFICATION AS REGRESSION

4.1 Preprocessing of Raw Pixel Features

We plan to use raw pixel image features as our input to the learning algorithms. Given that the input image has $424 \times 424 \times 3 = 539,328$ dimensions, it will be very difficult to learn from the limited training examples due to the curse of dimensionality. However, given the characteristics of the galaxy images, with most energy centered in the middle of the image, we are able to crop and resize the images to reasonable dimensions and still process them as raw pixel features.

4.1.1 Center-cropped Resized Dataset

We choose the dimensions of the RGB image to be $32 \times 32 \times 3 = 3,072$. The size is as small as possible, as long as we can still identify its class visually with our eyes; i.e., we try to keep the discriminative visual information (see Figure 4.1). As for the choice between 64×64 gray images and $32 \times 32 \times 3$ color images, we choose the latter after performing some test experiments, and the color images performed much better. Furthermore, an astronomy researcher informed us that color is an important feature for visually classifying galaxies.

4.1.2 Data Augmentation

We also applied a series of data augmentation techniques to fight the overfitting issue. We randomly flipped, rotated, and changed the color channels slightly for each training example. Feeding input data in this fashion, we have largely augmented our data and improved the later training results.

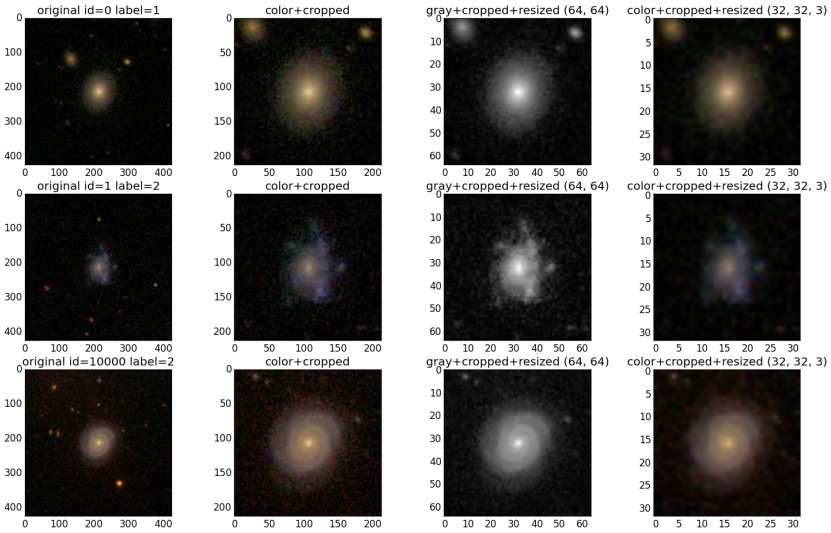

Figure 4.1: Examples of choices for preprocessing. We use the $32 \times 32 \times 3$ color images as our final input features for some subtasks.

4.2 Training the CNN

We divided the dataset into 80% training images and 20% validation images. All data was sealed in the form of pickled Python objects for fast processing of the neural network code.

Even at the high level of choosing neural network structure hyperparameters, the possibilities are enormous (see Figure 4.2). We carried out experiments to search over about 40 different neural networks, based on the pre-trained parameters and layer structures provided with the code for the CIFAR-10 dataset [11], using different numbers and sizes of filters, different numbers of convolutional network layers, fully connected layers, pooling and normalization layers, etc.

The best neural networks in the search came with three convolutional layers with more $6 * 6$ sized filters, each followed by a max-pooling layer, and only one fully connected layer to output the 37 predictions, which then connected to the final sum layer to compare with the ground truth labels to compute the MRSE as the training objective.

This network is much smaller than that trained in the ImageNet ILSVRC-2012 dataset, referred to as AlexNet [20], because the dataset is much smaller

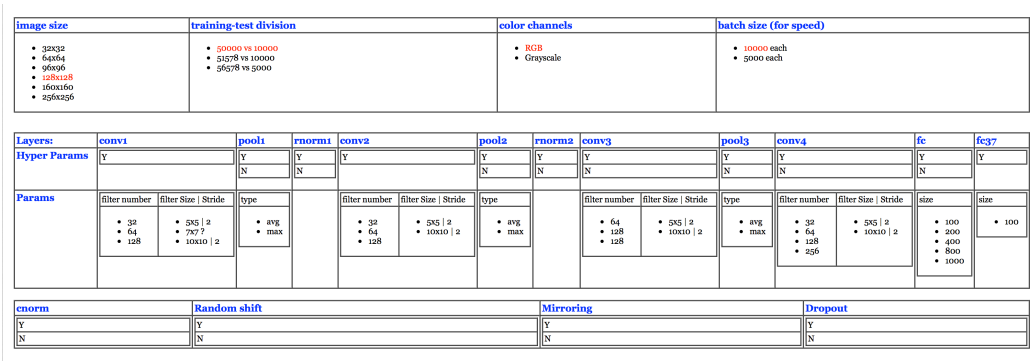

Figure 4.2: Scheme for searching the best neural network architecture

than the ImageNet, and the task is simpler compared to the 1000 class natural image classification task.

4.3 Results Post-processing

The predictions of the neural network were flattened 37-dimension vectors. The network did not utilize the correlation of the 37 classes. By applying boundary restraints at the final output, the final results (the RMSE calculated by submitting entries) can be improved by 4%. The restraints here are the boundary conditions of probabilities.

4.4 Final Results for the Regression Problem

The final result we achieved in this regression task is RMSE of 0.10090, which can be considered approximately 90% accurate.

Looking more closely at the CNN model, the learned filters from the first convolutional layer of the optimized convolutional neural networks are shown in Figure 4.3. The fitlers represent some of the low level visual patterns learned automatically and directly from the galaxy images, such as edges, corners, colors, etc. These patterns could capture and represent the characteristics of the galaxy images to a great extent but apparently are hard for people to handcraft and select, which again illustrates the feature extraction power of CNNs.

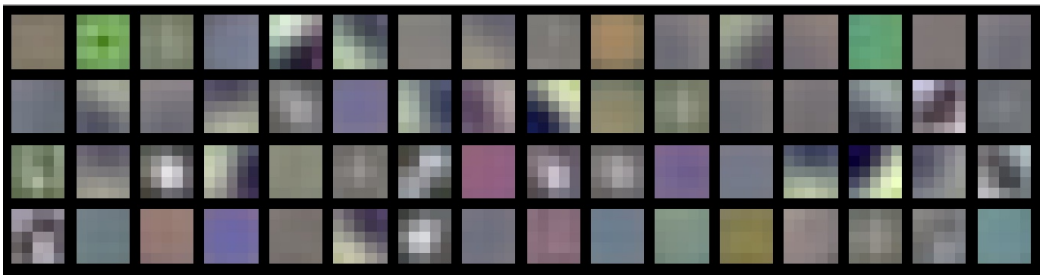

Figure 4.3: Filters from first convolutional layer

CHAPTER 5

GALAXY CLASSIFICATION AS MULTI-CLASS CLASSIFICATION

5.1 Preprocessing for Classification

In contrast to the regression task, for multi-class classification we need to preprocess both the galaxy image data and class labels for the task.

5.1.1 Preprocessing Galaxy Image Data

For the classification task, we also apply center-cropping, resizing and data augmentation techniques to the galaxy image data. In addition, we applied dimensionality deduction algorithms to perform extra experiments.

Principal component analysis (PCA) [21] is a widely used unsupervised dimension reduction algorithm. PCA converts a set of observations into a set of linear uncorrelated "principal components" where the number of such components is generally less than the number of original features.

In our implementation, we generated a reduced dataset from the $32 \times 32 \times 3$ color image dataset and performed experiments with some of the classification algorithms. In the choice of number of principal components, we used the criterion of keeping 99% of the variance calculated during the decomposition, and our final number of features is $1,143$.

5.1.2 Preprocessing Galaxy Class Labels

There are two problems making the galaxy classification subtle and difficult: (1) the intrinsic ambiguity of galaxy classes, and (2) the noisy labels from the dataset and our lack of ground truth labels.

Currently, there are two basic ways to classify the galaxies. The first method is based either on the classic T-types catalogue [3], or on the mor-

phological index T [2], which is a numerical index of a galaxy's stage along the Hubble sequence. The second method is based on shapes and structures as in the GalaxyZoo Project 2. Volunteers answer multiple-choice questions regarding the shapes and features (see Figure 2.2) of the given images, and the website generates the final result from all collected answers by voting and some post-processing [12],[5]. The intrinsic ambiguity of the galaxy classes here comes from the fact that some galaxy images are very likely to rank between two nearby classes in the catalogue, or have very similar appearance so that it is difficult to decide which class they belong to (for example, some galaxy images might be exactly between T-type "E0" and "S0", or equidistant between "smooth" and "having a disk or feature").

Another issue is the trustworthiness of the label. There are on average 44 people to classify one image; when the image has gone down all four levels, there might be only a few people on the last level to make the classification. Thus, the error here could have introduced more uncertainty than we can accept.

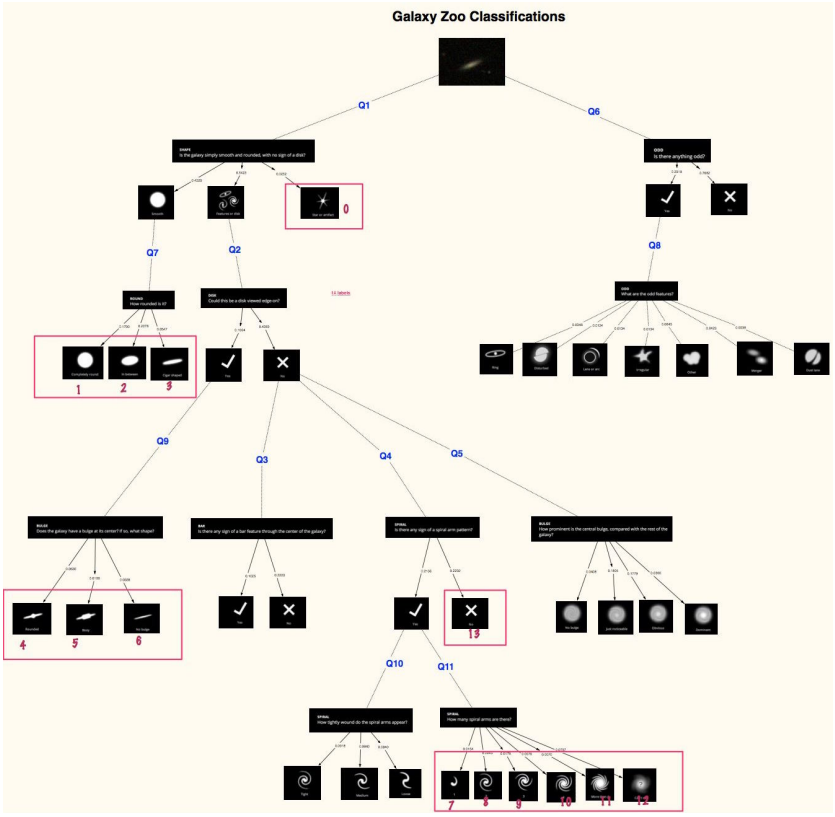

Figure 5.1: Illustration of the 14 labels choosing strategy

We finally generated the ground truth labels and prepared data in two ways. As illustrated in Figure 5.1, we chose 3, 6, or 14 labels of all training images (eliminating the labels with father-son relationship, so the 3, 6, or 14 probability labels are mutually exclusive and add up to 1). The 3-class problem is also reduced to a binary classification problem due to only very few examples from the 3rd class. And we chose only the images with certain trustworthiness (probability above a certain threshold to be some class).

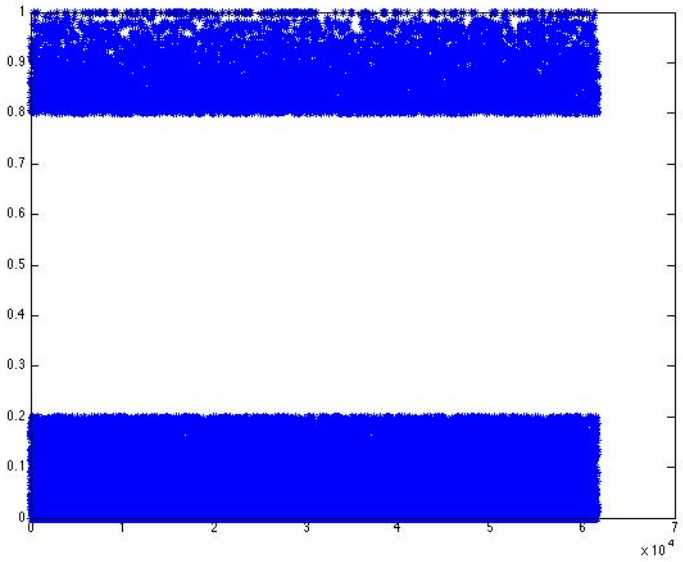

Figure 5.2: Cleaned data with threshold P(P=0.8 here) for the reduced 2-class classification task. Each data point represents an example. Dots in the upper blue region mean "smooth" and in the lower blue region "with features or disk."

We also divided the dataset into strips of certain probability regions to represent the different levels of trustworthiness (or different levels of classification difficulty), similar to what is illustrated in Figure 5.2.

5.2 Learning the Classification Model

For the classification task, we implemented both convolutional neural networks and conventional machine learning techniques.

To adapt the CNN model from the regression task to the classification

task, instead of using a fully connected layer to generate a real-valued vector as final prediction, we used a softmax layer following fully connected layers' input components, x_i's, to generate the probability of predictions for different classes.

$$softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{5.1}$$

And instead of using RMSE as loss function, we used cross entropy as loss. As in equation 5.2, y' is ground truth and y is the prediction.

$$H_{y'}(y) = -\sum_i y'_i \log(y_i) \tag{5.2}$$

For some of the conventional methods, we used both raw pixel features and PCA features as input, and results are discussed in the following section.

5.3 Final Results for the Classification Problem

5.3.1 Results for the Multi-class Classification

For the 6-class classification task, we achieved around 70% classification accuracy.

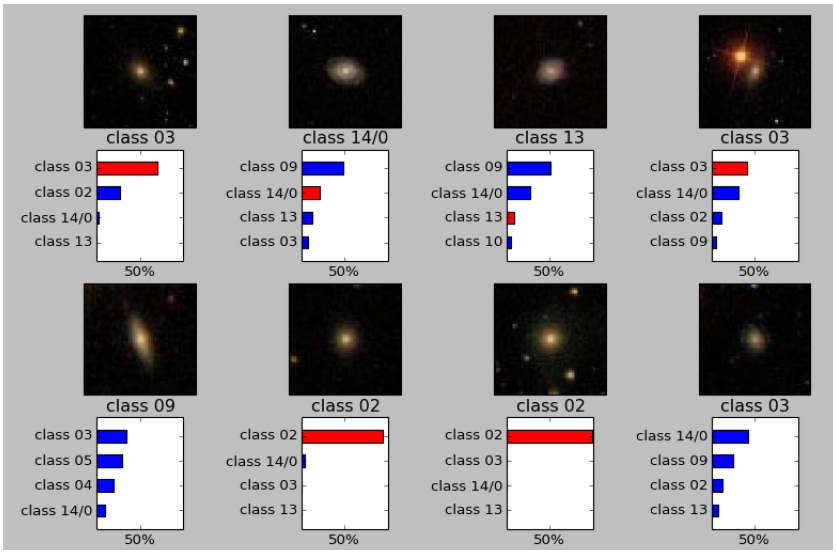

Figure 5.3: Predictions for the 14-class experiment with 59% accuracy

For the 14-class case with $64 * 64$ sized training images, we achieved 59% accurate classification rate. Some random predictions for the test dataset are in Figure 5.3. The length of the bar below each image represents the confidence of the prediction being a certain class, and red means correct prediction.

5.3.2 Results for the 2-class Classification

Using $32 \times 32 \times 3$ raw pixel features, we achieved relatively high classification accuracy on the test set for the "smooth galaxy" versus "galaxy with feature or disk" 2-class classification task. As shown in Figure 5.4, most of the results are between 0.8 and 0.9 except for naive Bayes and random forest. Notice that using PCA features does not reduce the classification score in the experiments due to the highly concentrated energy in galaxy images.

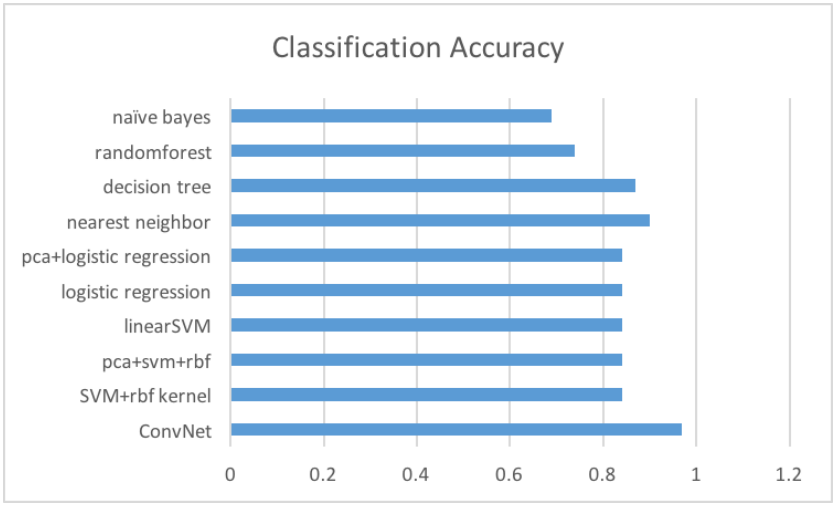

Figure 5.4: Comparison of classification accuracy for different algorithms

The highest classification accuracy of 97.7% is achieved by convolutional neural networks, which is significantly better than previous automated algorithms using handcrafted morphological features that are in general below 88% and even $2\% - 7\%$ better than astronomy experts in the sense of reproducing crowdsourcing classification results [5]. An example of prediction using CNNs with random picked test data from the original dataset is shown in Figure 5.5.

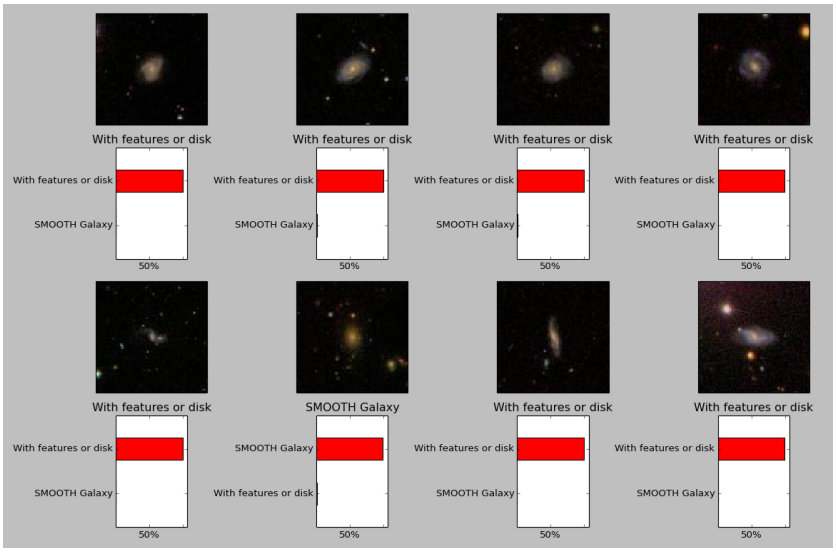

Figure 5.5: Predictions for the 2-class experiment with 97.7% accuracy

5.3.3 Training and Testing Time Efficiency

As we mentioned earlier, the training time and testing time are implementation oriented. However, we can still get some insights from the results.

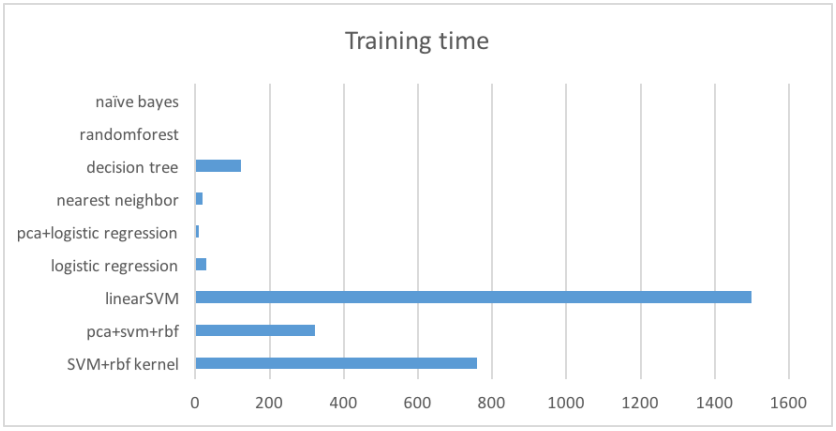

Figure 5.6: Comparison of training time for different algorithms

The training times (or setup time in the case of nearest neighbors) for naive Bayes, random forest, decision trees, nearest neighbor, and logistic regression are much shorter than that of SVMs, as shown in Figure 5.6. Here, linear SVM trains much more slowly than RBF kernel SVM; however, this is because the underlying library is not optimized for linearSVM. And

algorithms using PCA features are much faster due to the lower feature dimensionality.

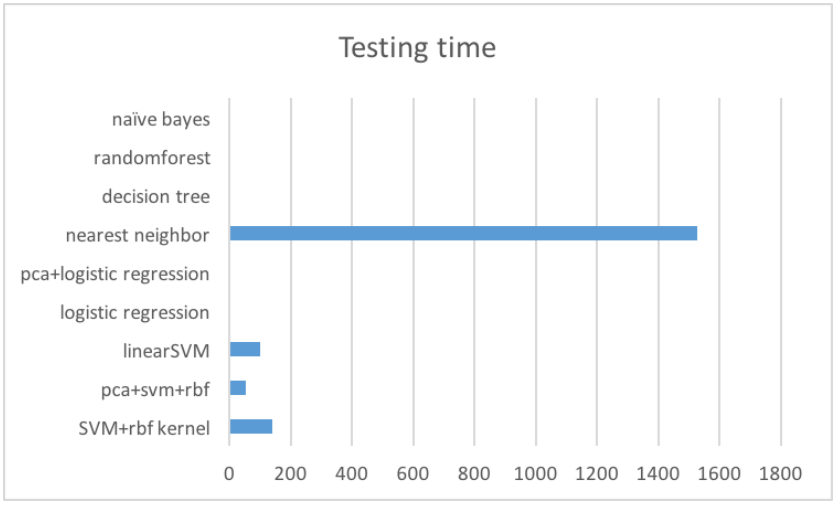

Figure 5.7: Comparison of testing time for different algorithms

For the testing time, shown in Figure 5.7, as we would expect, nearest neighbor method takes the longest time to calculate all the pairwise distances, and SVMs still take much longer than the rest of the algorithms except for nearest neighbor.

5.3.4 Comparison of Results to Human Learning

We also experimented with the trained convolutional neural networks on the different difficulty data. The results are in Table 5.1. Testing results with training set from $[0.8, 1.0]$ are listed here, but similar results are achieved for training set from $[0.5, 1.0]$, $[0.5, 0.6]$, $[0.9, 1.0]$).

The results are interesting in this way: All learned neural networks testing at different difficulty levels performed well at the tasks humans are good at, and badly at the tasks humans are bad at. If adding the data not listed here, another interesting comparison is that learning over the easy ones (training set in $[0.9, 1.0]$) with good understanding level (with 97.6% validation accuracy) produces slightly better results overall than learning the difficult one (training set in $[0.5, 0.6]$) at an average understanding level (with 59.5% validation accuracy). In this sense, artificial neural networks do have some sort of high level intelligence similar to that of humans.

Table 5.1: TEST with different difficulty levels

Test difficulty level(P)	Test with learning from $[0.8, 1.0]$
$[0.8, 1.0](validation)$	0.023(test error)
$[0.5, 0.6]$	0.390
$[0.6, 0.7]$	0.200
$[0.7, 0.8]$	0.100
$[0.8, 0.9]$	0.025
$[0.9, 1.0]$	0.008

CHAPTER 6

CONCLUSION

Conventional galaxy classification methods take advantage of carefully hand-designed morphological features, but our experiments show that preprocessed raw pixel features are also discriminative enough with state-of-the-art machine learning classification algorithms. However, feature extraction is still a key underlying step in galaxy image classification tasks, and convolutional neural networks can automatically learn more meaningful and representative features from the raw pixels to improve the classification performance. Experimental results show that an optimized CNN model with its hierarchical representation produced significantly better results than conventional methods and is suited for studying astronomical galaxy images.

REFERENCES

[1] Sloan Digital Sky Survey: Data Release 7. (Accessed Dec. 2014.) [Online] Available: http://www.sdss.org/dr7.

[2] M. Fukugita, O. Nakamura, S. Okamura, N. Yasuda, J. C. Barentine, J. Brinkmann, J. E. Gunn, M. Harvanek, T. Ichikawa, R. H. Lupton, D. P. Schneider, M. A. Strauss, and D. G. York, "A catalog of morphologically classified galaxies from the Sloan Digital Sky Survey: North equatorial region," *Astronomical Journal*, vol. 134, no. 2, pp. 579–593, Aug. 2007.

[3] P. B. Nair and R. G. Abraham, "A catalog of detailed visual morphological classifications for 14034 galaxies in the Sloan Digital Sky Survey," *arXiv.org*, Jan. 2010.

[4] A. Baillard, E. Bertin, V. de Lapparent, P. Fouque, S. Arnouts, Y. Mellier, R. Pello, J. F. Leborgne, P. Prugniel, D. Makarov, L. Makarova, H. J. McCracken, A. Bijaoui, and L. Tasca, "The EFIGI catalogue of 4458 nearby galaxies with detailed morphology," *Astronomy & Astrophysics*, vol. 532, p. A74, Aug. 2011.

[5] C. Lintott, S. Bamford, K. W. Willett, E. Edmondson, K. L. Masters, B. D. Simmons, K. R. V. Casteels, L. F. Fortson, S. Kaviraj, W. C. Keel, T. Melvin, R. C. Nichol, M. J. Raddick, K. Schawinski, R. J. Simpson, R. A. Skibba, A. M. Smith, and D. Thomas, "Galaxy Zoo 2: detailed morphological classifications for 304,122 galaxies from the Sloan Digital Sky Survey," *arXiv.org*, Aug. 2013.

[6] C. Lintott, S. Bamford, K. Schawinski, A. Slosar, K. Land, D. Thomas, E. Edmondson, K. Masters, R. C. Nichol, M. J. Raddick, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg, "Galaxy Zoo 1: data release of morphological classifications for nearly 900000 galaxies," *Monthly Notices of the Royal Astronomical Society*, vol. 410, no. 1, pp. 166–178, 2011.

[7] E. Bertin and S. Arnouts, "SExtractor: Software for source extraction," *Astronomy & Astrophysics Supplement Series*, vol. 117, no. 2, pp. 393–404, June 1996.

[8] M. Banerji, O. Lahav, C. Lintott, F. B. Abdalla, K. Schawinski, S. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar, A. Szalay, D. Thomas, and J. Vandenberg, "Galaxy Zoo: Reproducing galaxy morphologies via machine learning star," *Monthly Notices of the Royal Astronomical Society*, vol. 406, no. 1, pp. 342–353, 2010.

[9] M. Huertas-Company, J. A. L. Aguerri, M. Bernardi, S. Mei, and J. Sanchez Almeida, "Revisiting the Hubble sequence in the SDSS DR7 spectroscopic sample: A publicly available Bayesian automated classification," *Astronomy & Astrophysics*, vol. 525, Jan. 2011.

[10] Y. LeCun, Y. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of IEEE*, pp. 2278–2324, 1998.

[11] A. Krizhevsky, "High-performance C++/CUDA implementation of convolutional neural networks." (Accessed Apr. 2014.) [Online] Available: https://code.google.com/p/cuda-convnet/.

[12] GalaxyZoo Project. (Accessed Dec. 2014.) [Online] Available: http://www.galaxyzoo.org.

[13] GalaxyZoo: The Galaxy Challenge. (Accessed Dec. 2014.) [Online] Available: https://www.kaggle.com/c/galaxy-zoo-the-galaxy-challenge.

[14] T. P. Minka, "A comparison of numerical optimizers for logistic regression," unpublished draft, 2003.

[15] J. Su and H. Zhang, "A fast decision tree learning algorithm," *AAAI Conference on Artificial Intelligence*, vol. 6, pp. 500–505, 2006.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[17] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[18] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[19] D. E. Rumelhart, H. Geoffrey E, and W. Ronald J, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[20] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Conference on Neural Information Processing Systems (NIPS)*, 2012.

[21] K. Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.