Two Machine Learning Tasks on Spark platform

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Facing the era of data explosion in which massive data are created, stored and analyzed, Spark comes as a new framework for distributed computing. Spark provides native functional APIs that support multiple programming languages. In this paper, we elaborate on two machine learning experiments, facial keypoints detection and large scale video classification, conducted on Spark platform. Different models are deployed and compared based on performance. Through pratical experience of programming on Spark, we get familiar with concepts like resilient distributed datasets (RDD), etc. and detailed mechanism of distributed computing.

1 Detect the location of keypoints on face images

1.1 Introduction

Face detection is a classical and essential problem in computer vision and pattern recognization. It tackles gender regconization, emotion recognization and other real-life tasks that commonly occur in facial expression analysis, object tracking, etc. The challenges mainly come from the variety of illumination, rotation and pose. In this part, we propose a facial keypoints detection question retrieved from `www.kaggle.com`. We try to predict the locations of points including eyes, nose and mouth from 96×96 pixels of an image. We see it as a regression problem so as to fit a regression model on the pixel features. For simplicity, we call off-the-shelf interfaces in Spark library to complete this work.

1.2 Data preprocessing

Noticing there are many data with value NA in the location columns, we fill them with the average of the corresponding column. This may not be reasonable, but if we drop all instances with missing data, only 30% of training instances will be left. Besides, we scale the range of pixels to $[0, 1]$ by zero mean normalization. Also, input files are reorganized so that they can be transformed to RDD properly.

As for feature extraction, initially we sample 3,000 dimensions out of 9,216 but the computer fails to give results and throws memory error. Therefore, we decide to use 512-dimension pixel features from random sampling at last. In addition, we deploy PCA for dimension reduction. Note that SVD upon such a matrix is infeasible so we implement PCA over the sampled features.

1.3 Models

In this section we briefly illustrate some regression models employed in the experiments later.

Submitted to 30th Conference on Neural Information Processing Systems (NIPS 2016). Do not distribute.

Isotonic regression Isotonic regression is a generalized form of linear regression that fits a non-decreasing funtion to data. The free-form property of isotonic regression makes it outperform linear regression sometimes because it fits the data points in a more elastic way.

Tree-based models Tree-based regressions are all in relation to decision trees to some extent. In particular, gradient boosted tree regressor uses gradient boosting iteration to fit a decision tree. However, it can hardly be parallelized due to the sequential nature of boosting. Random forest regressor outputs the mean prediction of individual trees, which avoids overfitting habits of decision trees.

1.4 Experiments

Since we do not have the ground truth of the test data, we randomly split the training set by $9 : 1$ for training and testing repectively. The metric we use to evaluate the results is root of mean squared error (RMSE), i.e. the root of average deviations between the true location coordinate and the prediction.

We load the data into RDD and transform features into vectors so they can be valid arguments of models in Spark. Then we call different modules from the library `pyspark.ml.regression`. For each model, we first construct a model instance and fit it over the training data. After that we make predictions by calling `transform()` method. The predictions are evaluted by evaluator in `pyspark.ml.evaluation`.

To compare models, we take "left_eye_center_x" as example and show the results in the following table.

Table 1: RMSE of predictions

Model	RMSE
Linear regression	3.56
Decision tree	3.80
Random forest	3.35
Gradient boosted tree	3.62
Isotonic regression	3.54

1.5 Discussion

In Table 1, random forest regressor gives the best result. It outperforms decision tree just as expected, indicating overfitting may occur in the latter case. Gradient boosted tree regressor is said to be able to handle data of mixed type, which cannot be ratified in this experiment. Linear regression, as a basic method in regression, yields intermediate performance among all. Here we observe a subtle but significant gap between isotonic regression and linear regression. If we look into the pairs of the true value and prediction, we can see in some cases the error is relatively small (< 1). This implies there are cases in which our regression models do not work well.

Considering loss of information due to feature dimension sampling and training instance sampling, we can expect better performance with more efficiency of data usage. On the other hand, we can introduce features which are more sophisticated, such as pretrained CNN features. Meanwhile, efforts can also be made on a technical level, i.e. utilizing neural network models instead of basic regression models.

2 Large-scale video classification

2.1 Introduction

Video is one of the most common forms of multimedia nowadays. Videos are posted, transferred and viewed throughout the Internet and users' terminal devices. Video classification is a technique that attributes videos of similar theme or content to one category. It is widely applied in content search and recommendation of online videos.

For efficient classification, multiple features should be taken into consideration. Static appearance information and acoustic channels, motion clues are mutually complementary in representing a video. As for the former, CNN based representations are usually selected as static features. Motion features extends frame-based local features into 3D space. One can locate densely sampled frame patches to generate dense trajectories.

We work on Fudan-Columbia Video Dataset (FCVID) which contains over 90k Internet videos with 239 manually annotated categories. We utilize classification methods including naïve Bayes, multi-class logistic regression, etc. and compare the prediction accuracy. We also study inter-class relationships which are said to boost classification performance but also cause confusion. For example, class "WeddingReception" and "WeddingDance" may be correlated.

2.2 Features

CNN CNN is proved successful on image feature extraction in which pixels are filtered by feature map. However, its performance in video classification meets the bottleneck caused by complexity and scale of videos. Works have been done to extend the CNN to exploit more information, such as spatial-temporal space (Ji *et al.*, 2010), two-stream CNN (Simonyan *et al.*, 2014) and some advanced feature encoding strategies (Xu *et al.*, 2015).

HOG Histogram of oriented gradients (HOG) is a feature descriptor that counts occurrences of gradient orientation in localized portions of a video frame. It is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization to improve accuracy. The first step of descriptor calculation is computing gradients. The next step is to create the cell histograms. Every pixel within the cell delivers a weighted vote for an orientation-based histogram channel based on the values in the gradient computation.

SIFT Scale-invariant feature transform (SIFT) is a descriptor to detect local features in images or video frames. It is able to extract features in a way that reduces contribution of errors brought by image scale, noise and illumination. SIFT keypoints of objects are first generalized from a set of reference images. An object is recognized in a new image by comparing each feature from the new image to the forementioned candidates and finding cluster based on Euclidean distance of feature vectors. The clusters are determined by Hough transform.

2.3 Experiments

We obtain features from `http://bigvid.fudan.edu.cn/data/fcvid/FCVID_Feature/`. For each feature, we sample 512 dimensions. Likewise, we randomly sample 5,000 training intances among all 91,223. The ratio of training and testing data is 9 : 1. The evaluation metric is accuracy. Our experiments can be divided into the following parts:

We first apply different classifiers provided by `pyspark.ml.classification` library on the training set. We examine the performance of models by comparing accuracy of predictions they make based on the testing set, using `MulticlassClassificationEvaluator` in `pyspark.ml.evaluation`.

Then we check how features work complementarily. Features are deployed both individually and together to see whether feature fusion guarantees improvement in performance. The fusion method we use is simple concatenation of feature vectors.

Next, we change the scale of the training set in order to observe the effect on performance due to sample size. The number of instances ranges from 450 to 2,250. This part of experiment is implemented over multiclass logistic regression model.

Finally, we study inter-class relationships to mining knowledge sharing in classes.

2.4 Results

We display the comparison of classifiers in the table below.

Table 2: Accuracy of predictions (training intance: 1800)

Model	Accuracy
Logistic regression	0.31
Decision tree	0.07
Naïve Bayes	0.33
Random forest	0.06
Multilayer perceptron	0.17

In terms of features, we test different feature combinations.

Table 3: Features and accuracy (model: logistic)

Feature(s)	Accuracy
CNN	0.29
HOG	0.27
SIFT	0.26
CNN+HOG	0.30
CNN+SIFT	0.32

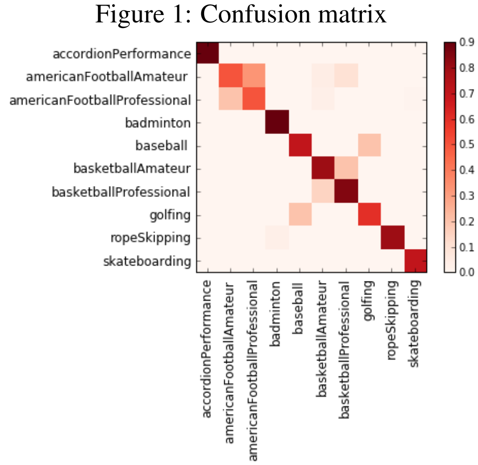
As training instances gradually increase , the accuracy keeps going up as Table 3 shows.

Table 4: Sample size and accuracy (model: logistic)

Sample size	Accuracy
450	0.15
900	0.27
1,350	0.30
1,800	0.31
2,250	0.37

Inter-class relationships are visualized with the confusion matrix below.

Figure 1: Confusion matrix



2.5 Discussion

It should be mentioned at the very beginning that two very low values in Table 2 result from the lack of training instances. Once we train these two tree-based models on a larger set, we can observe improvement of accuracy immediately.

Two relatively simple models, naïve Bayes and multiclass logistic regression work effectively even with a small training set. The MLP and tree-based methods need more instances to learn a better model. Moreover, the latter may incur overfitting because we do not go through the pruning process.

About feature fusion, it is verified that feature combinations are able to promote the prediction accuracy. In fact, HOG supplements information in motion because CNN features are usually considered as static presentations of an image. Besides, SIFT carries audio descriptors that help categorize video semantics. Therefore, the three features we employ in this experiment contain static, motion and audio information respectively and are complementary to each other.

At last, we comment on Figure 1. The colored spots which do not lie on the diagonal indicate that the corresponding two classes are correlated while may bring confusion in classification. The knowledge sharing property is identified by matrix Ω (Jiang *et al.*, 2015).

References

[1] Y. Jiang, Z. Wu, J. Wang, X. Xue & S. Chang (2015) Exploiting feature and class relationships in video categorization with regularized deep neural networks. In *ArXiv preprint*.

[2] S. Ji, W. Xu, M. Yang & K. Yu (2010) 3d convolutional neural networks for human action recognition. In *ICML*.

[3] K. Simonyan & A. Zisserman (2014) Two-stream convolutional networks for action recognition in videos. In *NIPS*.