Document of Project

Zhangjie Cao
2014013428
caozhangjie14@gmail.com

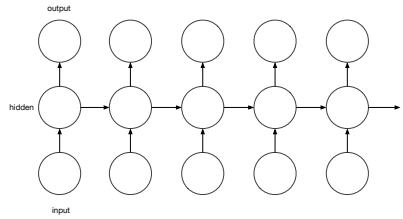
Task 1

Method

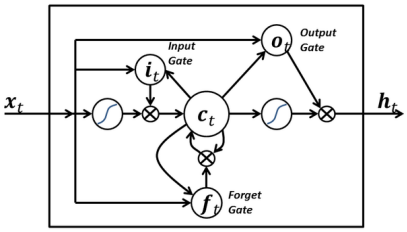
I use two methods in this task, where one is a deep method and another is a shallow method.

Shallow Method I design the traditional method with the well-known design procedure, which consists of feature extraction and data classification. As the raw data is difficult to classify or is not suitable to the classifier, some feature extraction method needs to be applied such as Fourier transform on time series data and SIFT on image data. Using method, we can transform data to points in a space which is easier to classify, such as map data into a linearly separable space. Thus, in the second step, we only need some traditional classification method which is only capable of tackling convex or slightly non-convex problems. In this task, I use Discrete Fourier transform (DFT) to map the time series at every dimension of data to its corresponding frequency space.The time series data where each timestamp is depend on its previous and next timestamp and this dependency is difficult to model. However, in the frequency space, this dependency doesn't exist and every dimension is a response of one component of some frequency. So in the second step, I only need the simple linear classifier to classify the data. I use stochastic gradient descent (SGD) to train the model.
Support vector machine (SVM) is a widely used method in shallow learning. It minimizes the classification error and at the same time maximizes the margin between the data and the classification bound. Thus I also try this method as comparison, although this is not the rigidly defined time series classifying approach. I use the time series of 20 as a training point in the SVM. As the data dimension is too high, I use PCA the reduce the dimension of data.//

Deep Method The deep method is based on LSTM (Hochreiter and Schmidhuber 1997), which is a widely used recurrent neural network (RNN) architecture. Compare to the simple RNN which is shown in Figure 1(a), it has more complex structure in the transformation at every timestamp. As shown in Figure 1(b), LSTM has a memory unit to store and update information throughout the propagation of input along the time. And there are some gates to control the information flow. The forget gate $\mathbf{f_t}$ can block the information from the last timestamp. The input gate $\mathbf{i_t}$ control how much the input data is integrated into the hidden state. The output gate $\mathbf{o_t}$ control how much the output at the current timestamp is influenced by the hidden state. Thus, with the hidden state $\mathbf{c_t}$ storing the information through the whole time series and the gates controling information flow, LSTM performs better than tradition RNN on most tasks based on series data. With some modification to fit the particular task, LSTM has achieved state of the art performance on various tasks. The hidden state dimension is set as 10 in the experiment.
As the task just a classification problem with high dimension data. I also try the easiest deep model, which is multi-layer perceptron in the experiment. It's just a two-layer perceptron with the second layer being the classifier. As the data dimension is too high, I use PCA the reduce the dimension of data as in SVM.// I also try the deep convolution network in the experiment. As the input is not normal 2D image but a high-dimension time series. It's not naturally to use



(a) Recurrent Neural Network



(b) LSTM

Figure 1: The simple recurrent neural network architecture and LSTM.

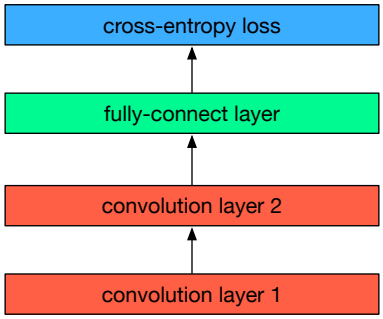


Figure 2: My Architecture of Convolution Neural Network

normal 2D filter on this data. Thus, I use $1 * kernel\ height$ kernel to slide on the time dimension and the input channels is set as the feature dimension. Thus there are $input\ channels(feature\ dimension) * output\ channels$ kernels in this convolution layer, followed by a relu activation layer, a max pooling layer and a drop layer. The second convolution layer is designed similarly only with different dimensions. On these two convolution layer, a fully-connected layer is applied as the classifier. The overall architecture is shown in Figure 2

Result

Basically, there are two evaluation methods well known in the classification society. One is accuracy and another is Precision-Recall and F-measure. However, these two methods only differs in the situation where positive and negative data instances are imbalance. But in our data, the positive and negative data instances are nearly the same. Thus, these two evaluation methods measure the same thing. We need only compare one evaluation methods and I select accuracy. In addition, I also compare the time performance of these two methods. // For deep method. nn achieves an accuracy of 91.78%, CNN achieves an accuracy of 60.80% and LSTM achieves an accuracy of 60.60%. For shallow learning method, Fourier Transfer method achieves an accuracy of 24.36% while SVM achieves an accuracy of 94.51%. We can see that the simpler model performs better than other models for deep learning method. This reflects that the hidden structure of the data is not very complex and complex model is tend to overfit. The incredible performance of SVM also emphasize this point. On the other hand, PCA plays a important role in the learning. PCA reduces the dimension of data, which makes the training easier and at the same time have very little loss. Thus, nn and SVM performs better than other methods.

Conclusion

In this experiment, I use the five methods to classify the data and compare the performance of these five methods. The accuracy performance varies much and comparing these methods, I draw some conclusion about the data and find the strength and shortcoming of different method. It's a good practice on classification.

Task 2

Method

I use the K-means clustering method in this task. K-means is an unsupervised vector-based clustering algorithm that is widely used in machine learning, computer vision and data mining community. It clusters all the data points into k different clusters, where k representing the number of clusters, is the only one hyper-parameter of K-means. The k should be decided by the user before running K-means.

K-means aims to cluster every data point to one of the k nearest mean points, where each mean point represent a cluster. The problem is computationally difficult (NP-hard), but there is a heuristic algorithm which is commonly employed and converges quickly to a local optimum. It process as follows:
(1) It initializes k means $m_1, m_2, ..., m_k$. **(2)** It assigns each data point to cluster of the nearest mean. **(3)** It computes the mean of data points in each cluster and assigns new means to the corresponding cluster. **(4)** The algorithm loop the **(2)** and **(3)** step until the assignment of **(2)** is no longer changed.

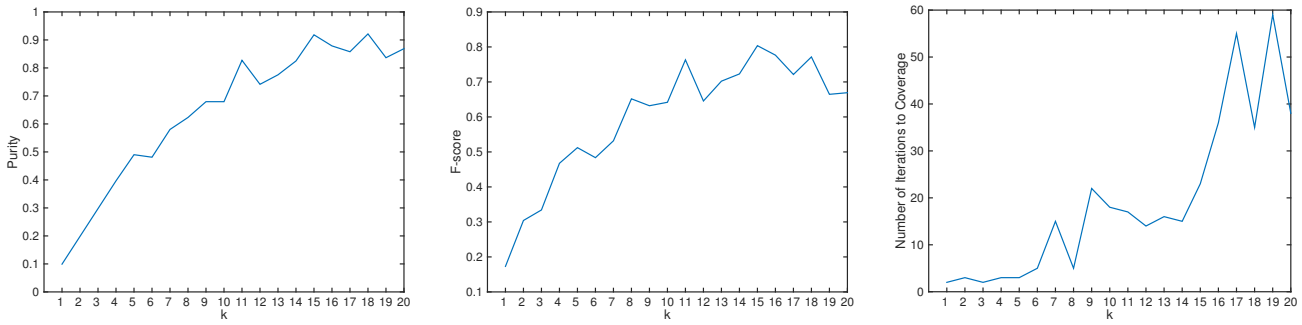
This heuristic algorithm is also called Lloyd's algorithm. The running time of this algorithm is $O(nkdi)$ where n is the number of data points, d is the dimension of each data points, which is a vector, k the number of clusters and i the number of iterations needed until convergence.

Results and Analysis

I evaluate the effectiveness of K-means with purity and F-score. Purity reflects how much each cluster is mainly directed by one class. It first calculates accuracy by dividing the number of most frequent class (defined by ground truth label) in each cluster by the number of all data points in each cluster, and then gets the purity by the weighted accuracy where the number of all data points in one cluster is regarded as the weight of this cluster. F-score reflects quality and completeness of clustering. Pairs of data categorized into one cluster under ground truth label are defined as a set $\mathcal{S}$, and pairs of data categorized into one cluster under the result of our algorithm are defined as a set $\mathcal{T}$. The F-score is the harmonic mean of pair precision and pair recall, where pair precision means $\frac{|\mathcal{S}\cap\mathcal{T}|}{|\mathcal{T}|}$ and pair recall means $\frac{|\mathcal{S}\cap\mathcal{T}|}{|\mathcal{S}|}$.

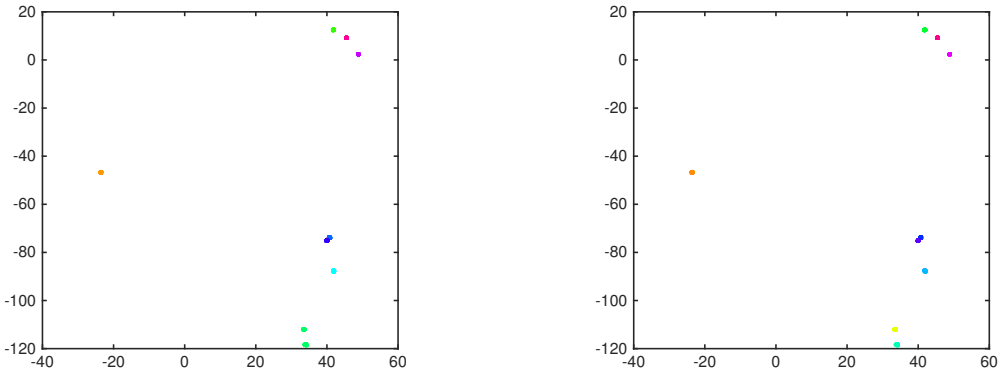
Figure 3 show the results under different evaluation methods with respect to k.

We can see from Figure 3(a) that with the increasing of k, purity is increasing too but when k is larger than 15, purity doesn't increase and is balanced around a value. Since all the data points are divided into more and more clusters, the number of points in each cluster tends to decease and the data points in one cluster tends to become closer. Data points in one cluster tends to have the same ground truth label. Thus, purity is increasing. When k is large enough, purity is very high and saturated, which means most of data points in one cluster are belongs to the same label. Once k are nearly the number of data points, purity is approximately 1.



(a) Purity (b) F-score (c) The Number of Iterations

Figure 3: The experiment results of Purity, F-score and Number of Iterations to coverage with respect to k.



(a) K-means (b) Ground Truth

Figure 4: The Visualization of Clustering for K-means and Ground Truth

We can see from Figure 3(b) that with the increasing of k, F-score is increasing first but when k is larger than 15, F-score is decreasing. Since all the data points are divided into more and more clusters, the number of points in each cluster tends to decease and the data points in one cluster tends to become closer, which indicates that the possibility for a pair of data in one cluster are truly close becomes higher. Thus the pair precision is increasing. However, the larger number of cluster also means some data pairs are truly close tends to be divided into two cluster. Thus the number of truly close pairs defined by the clustering result of K-means is much smaller than the number of all the truly close pairs. Thus, pair recall tends to decrease. There is a trade-off between the precision and recall. As the k is increasing, pair precision increases less while pair precision decreases more. When k is around 11, the real number of clusters, F-score is the highest. But as the random initialization. F-score @$k = 11$ is not the highest in Figure 3(b), but the highest value is F-score @$k = 15$, which is around 11.

In the previous section, I theoretically analysis the time complexity of K-means algorithm, which is $O(nkdi)$. In this complexity, only the number of iterations is not known before running K-means. So I study the number of iterations in the Figure 3(c). We can see that as k increase, more and more iterations are needed for K-means to coverage. As k is increasing, the mean of cluster becomes closer and a data point belongs to one cluster is more easier to belong to another cluster under the same change of mean. Thus the number of iterations to coverage is increasing.

Now, we compare the visualization of the result obtained from K-means and from ground truth in Figure reffig:vis we can see that K-means categorize data points into cluster similar to ground truth and only wrongly cluster two clusters of data into one, which indicates the effectiveness of K-means visually. Statistically, the largest F-score is 0.8 with the purity 0.9, which is very significant, proving the effectiveness of K-means further.

However, one disadvantage of K-means is that the best k is difficult to decide and the user need to start from a heuristic k and try some k to find the performance change with respect to k before getting the best k. And the heuristic algorithm only get the local optimum, which is not the best clustering result and sometimes may coverage to result that is not desirable.

Conclusion

In this task, I implement K-means algorithm to cluster localization data. From the result of clustering, I get some insight into K-means algorithm and evaluate this algorithm from different view. In the future, I may think of some novel

methods to improve the initializtion of K-means, which is important for the performance of Lloyd's algorithm.

References

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.