11775-SG: Homework 2

YongKyung Oh*
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
yongkyuo@andrew.cmu.edu

Abstract

The task of homework 2 is to perform multimedia event detection (MED) with video features. Main tasks are extract SURF and CNN features from video file and develop model for multiple events. For the SURF and CNN, Kmeans clustering is used to define and represent features. in CNN, Vector of Locally Aggregated Descriptors (VLAD) Encoding technique is implemented to extract additional features. Also, CNN features are used to classifier without representation step. SVM classifier and LGBM are developed for the baseline and customized classification approach is suggested.

1 Introduction

- The overview of MED pipeline is depicted in Figure 1. In the first step, we extract features from the raw training data. In this homework, the video feature SURF and CNN are used.
- Implement the bag-of-words representation with k-means clustering. SURF features are extracted by open CV (Which is provided). CNN features are extracted by Resnet 18. For Features, probability of each K-means cluster is used as representing features.
- Support Vector Machine (SVM) and Light Gradient Boosting Model (Light GBM; LGBM) are used to classify the video samples. Chi2-kernel is used. Additional to that, SMOTE oversampling is used to deal with imbalanced issue.

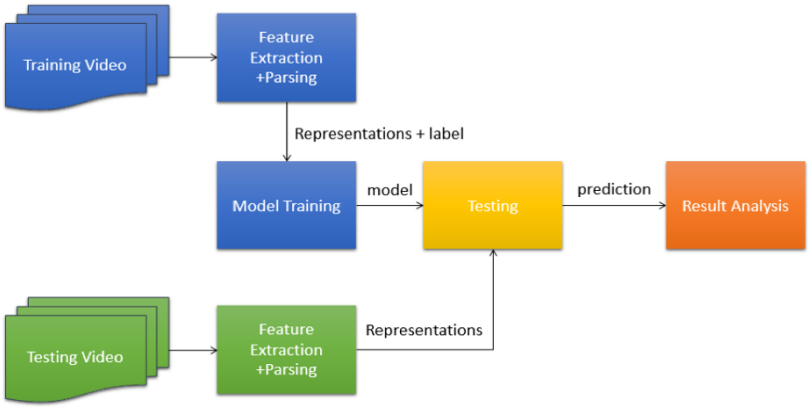

Figure 1: Project Pipeline

*UNIST, ok19925@unist.ac.kr

2 Data

The dataset contains 2935 videos, with 3 positive events (P001: assembling shelter; P002: batting in run; P003: making cake) and 1 negative event class (NULL).

- For training, the file **all_trn.lst** specifies 836 training videos and their labels.
- For validation, the file **all_val.lst** contains 400 videos and their ground-truth labels as well. Validation set is used to tune hyper-parameters.
- For testing, there are additional 1699 videos specified in the **all_test_fake.lst**, in which their labels are all fake (deliberately set as NULL)

2.1 SURF Features

The Speeded Up Robust Features (SURF) is a kind of hand-crafted local feature detector and descriptor, which was widely applied to tasks such as object recognition and image classification. The SURF descriptor could be viewed as an extension of the Scale-Invariant Feature Transform (SIFT) descriptor. The SURF algorithm first extracts key locations (interest points) of an image and then extracts visual attributes around these locations. Therefore the data structure for one video is [# *of Key frames* × # *of key locations* × # *of feature dimension*]. In our data, feature dimension is 64.

2.2 CNN Features

Feature learning with convolutional neural networks (CNNs) have become a hot topic in computer vision and many other fields. In this part, please select your favorite neural network tools to extract the image-level CNN features on videos' key-frames. Depending on the layer you pick from the neural network, you may choose to skip the feature encoding/clustering process. I use the hidden feature vector of resnet 18 model. The feature dimension is 512 for one frame. So, the data structure is [# *of Key frames* × # *feature dimension*]. Here is the architecture of Resnet 18 Model. [2]

Layer Name	Output Size	ResNet-18
conv1	$112 \times 112 \times 64$	7×7, 64, stride 2
conv2_x	$56 \times 56 \times 64$	3×3 max pool, stride 2
		$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
conv3_x	$28 \times 28 \times 128$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$
conv4_x	$14 \times 14 \times 256$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$
conv5_x	$7 \times 7 \times 512$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$
average pool	$1 \times 1 \times 512$	7×7 average pool
fully connected	1000	512×1000 fully connections
softmax	1000	

Figure 2: Resnet 18 Architecture

2.3 Bag-of-Words Representation

Representing a video using bag-of-words is one of the feasible approaches. To speed up the clustering process, you can choose only a small portion of the SURF vectors and CNN features(ex. randomly

[2] Resnet 18 Architecture

select 20% SURF and 50% CNN from each video). To represent the video, K-means clustering is implemented. To determine the number of K, sum of squared error and silhouette score are used.

3 Experiments

3.1 K-means clustering

- SURF vector data structure for one video is [# *of Key frames* $\times$ # *of key locations* $\times$ # *of feature dimension*]. In our data, feature dimension is 64.
- CNN feature data structure is [# *of Key frames* $\times$ # *of feature dimension*]. The feature dimension is 512 for one frame.

To speed up, 20% random sample dataset for SURF (select.surf.csv) and 50% random sample dataset for CNN (select.cnn.csv) are used to test and develop model. This train data has dimension (5704226, 64) for SURF and (61441, 512) for SURF . Most of all, K-means clustering is implemented. I tested K for 180 and 360 and selected 360 which has higher dimension (or more features). Additional to that, current development environment (EC2 t2.large) is not suitable for the large computation and not available for GPU computation. To reduce the computational burden, I use the mini-batch k means with batch size 50.

> The MiniBatchKMeans is a variant of the KMeans algorithm which uses mini-batches to reduce the computation time, while still attempting to optimise the same objective function. Mini-batches are subsets of the input data, randomly sampled in each training iteration. These mini-batches drastically reduce the amount of computation required to converge to a local solution. [1]

MiniBatchKmeans has little bit worse result than original Kmeans. Kmeans cannot handle larger dataset. For example, if I try the cluster number 180 using Kmeans, then memory error occurs. Instead of the accuracy (or precision) performance, I use the larger feature dimension to develop the better classifiers.

3.2 Vector of Locally Aggregated Descriptors (VLAD)

The Vector of Locally Aggregated Descriptors (VLAD) is technique to represent features. We accumulate the residual of each descriptor with respect to its assigned cluster. In simpler terms, we first match a descriptor to its closest cluster as in Bag-of-Words. Later, for each cluster, we store the sum of the differences of the descriptors assigned to the cluster and the centroid of the cluster. I implemented following steps.

- Generate K-means clustering from the selected data
- Find the centroid of kmeans
- Substract the original feature from the centroids of all clusters
- Flatten the array. Normalize it

For example in the CNN, data structure is [# *of Key frames* $\times$ # *of feature dimension*]. If we implement the VLAD steps, we can get the VLAD representation in the dimension of [# *of clusters* $\times$ # *of feature dimension*] = [$360 \times 512 = 184320$] for one frame. So, one video has a feature of [# *of Key frames* $\times$ 184320]. This is huge and I couldn't train the classifier due to the memory issue.

3.3 Support Vector Machine (SVM) classifier

Before develop classifier model, I conducted re-sampling pre-process to deal with imbalanced issue. For example, list of train data contain 836 video name and label. The imbalanced ratio are very high. Counter('P001': 36, 'P002': 36, 'P003': 36, 'NULL': 728). $0.0430 (= 36/836)$. In this binary classification case, model should developed by minor class features. Class weight and re-sampling technique can be considered. In this case, SMOTE oversampling is used to sample the minor dataset.

SMOTE: Synthetic Minority Over-sampling Technique. To illustrate how this technique works consider some training data which has s samples, and f features in the feature space of the data. Note that these features, for simplicity, are continuous.

To then oversample, take a sample from the dataset, and consider its k nearest neighbors (in feature space). To create a synthetic data point, take the vector between one of those k neighbors, and the current data point. Multiply this vector by a random number x which lies between 0, and 1. Add this to the current data point to create the new, synthetic data point. [2]

For the classifier, support vector machine (SVM) classifer is used. SVM is a discriminatory classification formally defined by the separation hyperplane. In other words, when labeled train data (supervised learning) are given, the algorithm outputs an optimal hyperplane to classify new data. To tune the optimal hyper parameters, following factors are considered. [3].

- Kernel: Specifies the kernel type to be used in the algorithm. Chi2-kernel is selected, because the chi2-kernel do some normalization in the kernel itself, so it's often better than other kernels, especially for the such a large scale data.
- C: Regularization parameter. The strength of the regularization is inversely proportional to C. Instead of default 1, 2.0 is determined by experiments. More penalty to find better prediction for minor set.
- Class weight: Set the parameter C of class i to *class_weight*$[i] * C$ for SVC. Balanced class weight is used. Sampling is conducted, so it may not considered as well.

3.4 Light GBM classifier

LightGBM is an open-source framework for gradient boosted machines. By default LightGBM will train a Gradient Boosted Decision Tree (GBDT), but it also supports random forests, Dropouts meet Multiple Additive Regression Trees (DART), and Gradient Based One-Side Sampling (Goss). [3]

Boosting train models sequentially that each model learns from the errors of the previous model. Starting with a weak base model, the model is repeatedly trained, each in addition to the previous model's predictions, producing a strong overall prediction. [4]

Light GBM is widely used to deal with larger data with faster speed and performance. Grid search cross validation is implemented. According to the multiple experiments, different parameter classifiers works well in the different event settings. Therefore, I conducted multiple grid search CV for each event. I consider following parameters: num_leaves, $min\_data\_in\_leaf$, $lambda\_l1$, and $lambda\_l2$.

4 Results

In case of SURF, the number of key location in a frame varies. Therefore thresholds are setup. SURF/100 model is selecting 100 key locations in a frame and SURF/1000 model is selecting 1000 key locations in a frame. CNN feature is used to represent as kmeans clustering and feature itself.

SVM model and pre-tuned Light GBM are used to compare the results. For CNN, the results are not consistent and raw CNN feature classifier works bad. So, I assume that there is a problem of extracting CNN feature from Resnet. So, I will try to implement different CNN model for the next assignment. The best cases are as follow using validation set (*all_val.lst*):

As expected, different model works well in the different events. In the case of SURF, feature dimension is not enough to improve the classifier performance. It is very hard to increase the k with current development environment, different approach may be required. I couldn't conduct further experiments due to memory issues.

[3] Implementation reference link for SVM

[4] Implementation reference link for Light GBM

		SURF/100	SURF/1000	CNN/Kmeans	CNN/Feature
SVM	P001	0.9250	0.9225	0.8850	0.9625
	P002	0.9675	0.9700	0.8175	0.9550
	P003	0.8875	0.8825	0.8450	0.9425
LGBM	P001	0.9550	0.9450	0.9525	0.9625
	P002	0.9725	0.9600	0.9425	0.9550
	P003	0.9350	0.9450	0.9225	0.9425

Table 1: Results: Accuracy

		SURF/100	SURF/1000	CNN/Kmeans	CNN/Feature
SVM	P001	0.1326	**0.1551**	0.0657	0.0375
	P002	0.3904	**0.4594**	0.1284	0.0450
	P003	**0.1107**	0.0992	0.0672	0.0575
LGBM	P001	0.0483	0.0461	0.0900	0.0375
	P002	0.4164	0.2790	0.1708	0.0450
	P003	0.0773	0.1283	0.0593	0.0575

Table 2: Results: MAP

Final score file is produced by using test set (*all_test_fake.lst*). SVM and Light GBM model is used to generate score file. For validation set, train data is used to generate model. For test set, train data and validation data are used to generate model. Total 48 files (2 data setting * 4 feature * 3 event * 2 model) are generated. For Best file is selected by the result of validation set. Every scripts (including VLAD) are submitted on the github.

References

[1] David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178, 2010.

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[3] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154, 2017.