D5.4: Sequential Algorithms

Mahesan Niranjan (University of Southampton)

Abstract.
FP7-ICT Strategic Targeted Research Project (STREP) ICT-2011-287863 TrendMiner
Deliverable D5.4 (WP5)

This deliverable describes a short piece of work carried out within the TrendMiner project, to work on algorithms that attempt to circumvent computational complexity by taking sequential, one-pass through the data. We focused on clustering problems, reviewed several related algorithms, and developed and evaluated a novel algorithm that sequentially partitions a given input space (SISC - Sequential Input Space Carving). An alternate algorithm we worked on is projective clustering, again achieving sequential clustering integrated with a subspace projection which can be computed efficiently. We show that the sequential algorithms are competitive in performance with batch method of k-means clustering in a task of constructing visual codebooks, but at vastly reduced computational costs. We further identify a direction for further work, beyond the scope of the project, in situations of small data in very high dimensions.

Keyword list: Sequential Algorithms, Cluster Analysis, Object Recognition, Coresets, Singular Value Decomposition

Project	TrendMiner No. 287863
Delivery Date	October 27, 2014
Contractual Date	November 1, 2014
Nature	Report
Reviewed By	Thierry Declerck (DFKI)
Web links	N/A
Dissemination	PU

TrendMiner Consortium

This document is part of the TrendMiner research project (No. 287863), partially funded by the FP7-ICT Programme.

DFKI GmbH
Language Technology Lab
Stuhlsatzenhausweg 3
D-66123 Saarbrcken
Germany
Contact person: Thierry Declerck
E-mail: declerck@dfki.de

University of Southampton
Southampton SO17 1BJ
UK
Contact person: Mahesan Niranjan
E-mail: mn@ecs.soton.ac.uk

Internet Memory Research
45 ter rue de la Révolution
F-93100 Montreuil
France
Contact person: France Lafarges
E-mail: contact@internetmemory.org

Eurokleis S.R.L.
Via Giorgio Baglivi, 3
Roma RM
00161 Italy
Contact person: Francesco Bellini
E-mail: info@eurokleis.com

Institute of Computer Science
Polish Academy of Sciences
5 Jana Kazimierza Str
01-248 Warsaw
Poland
Contact person: Maciej Ogrodniczuk
E-mail: Maciej.Ogrodniczuk@ipipan.waw.pl

Universidad Carlos III de Madrid
Av. Universidad, 30
28911 Madrid
Spain
Contact person: Paloma Martínez Fernández
E-mail: pmf@inf.uc3m.es

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Contact person: Kalina Bontcheva
E-mail: K.Bontcheva@dcs.shef.ac.uk

Ontotext AD
Polygraphia Office Center fl.4,
47A Tsarigradsko Shosse,
Sofia 1504, Bulgaria
Contact person: Atanas Kiryakov
E-mail: naso@sirma.bg

Sora Ogris and Hofinger GmbH
Bennogasse 8/2/16
A-1080 Wien
Austria
Contact person: Christoph Hofinger
E-mail: ch@sora.at

Hardik Fintrade Pvt Ltd.
227, Shree Ram Cloth Market,
Opposite Manilal Mansion,
Revdi Bazar, Ahmedabad 380002
India
Contact person: Suresh Aswani
E-mail: m.aswani@hardikgroup.com

DAEDALUS - DATA, DECISIONS
AND LANGUAGE, S. A.
C/ López de Hoyos 15, 3
28006 Madrid
Spain
Contact person: José Luis Martínez Fernández
E-mail: jmartinez@daedalus.es

Research Institute for Linguistics of the
Hungarian Academy of Sciences
Benczr u. 33,
1068 Budapest Hungary
Contact person: Tamás Váradi
E-mail: varadi.tamas@nytud.mta.hu

Executive Summary

This deliverable describes a short piece of work carried out within the TrendMiner project, to work on algorithms that attempt to circumvent computational complexity by taking sequential, one-pass through the data.

We focused on clustering problems, reviewed several related algorithms, and developed and evaluated a novel algorithm that sequentially partitions a given input space (SISC - Sequential Input Space Carving). An alternate algorithm we worked on is projective clustering, again achieving sequential clustering integrated with a subspace projection which can be computed efficiently.

We show that the sequential algorithms are competitive in performance with batch method of k-means clustering in a task of constructing visual codebooks, but at vastly reduced computational costs. We further identify a direction for further work, beyond the scope of the project, in situations of small data in very high dimensions.

Contents

Chapter 1

Introduction

This deliverable describes work carried out under `Task 5.4` of WorkPackage 5 in the TrendMiner project, and relates to an exploration of sequential algorithms. Our motivation is that, in computationally complex inference methods that cannot scale up to very large volumes of data, *sequential* algorithms that take *one-pass* through the data may well be the only practical way forward. Such sequential training of machine learning models is now an active topic of research, particularly using stochastic conjugate gradient and related methods to train inference models. Our work is closely related, but inspired by early work on the Resource Allocating Network (Platt, 1991), which was further developed in a function estimation perspective in Kadirkamanathan and Niranjan (1993) and shown to have demonstrable advantages in time series prediction problems (Kadirkamanathan *et al.*, 1992; Kadirkamanathan and Fallside, 1991). A closely related source of inspiration for this contribution is early work on sequential training of multilayer perceptron (MLP) neural networks, developed in Shadafan and Niranjan (1994). There, it was shown that, since a high-gain MLP applied to pattern classification problems essentially constructs a piece-wise approximation to the class boundary, it may be trained on local data by the application of the Recursive Least Squares algorithm.

In this task, we focus on clustering algorithms, building on the Resource Allocating Clustering (RAC) work developed in Southampton (Ramanan and Niranjan, 2012, 2010; Farran *et al.*, 2009). We extend RAC to derive a sequential one-pass algorithm we call Sequential Input Space Carving (SISC) approach to clustering. The resulting novel algorithm is shown to be capable of efficient clustering of data, evaluated on an image codebook construction task. The resulting codebook is compact, and can be designed in a fraction of the computing time it takes with batch k-means approach and the codebook results in no loss of image classification accuracy. We explore an alternate approach based on projections known as coresets. Here again, we show that sequential clustering is possible and can derive efficient clustering without significant loss of discriminating power.

While evaluations for this task have been carried out in the setting of codebook design for image classification, which is a convenient problem of large scale clustering, the algorithms explored are general and can be applied to any clustering problems of comparable scales.

Thus, in this task, we have

1. Explored a number of sequential algorithms for clustering, reviewed their properties, and developed a novel algorithm suitable for one-pass clustering. We have demonstrated that the novel algorithm can derive clusters that are competitive in performance in visual categorization tasks of which processing social media images would be a TrendMiner related application. It can achieve such performance at a vastly reduced computational costs.

2. We have identified a novel topic to develop beyond the TrendMiner project, working in a feature-sequential setting. This line of research is most appropriate for data in which the dimensions are much greater than the number of items of data.

Chapter 2

Sequential Clustering Algorithms

2.1 Sequential K-Means

The standard K-means algorithm, is amenable to parallel implementations. It is essentially an Expectation-Maximization algorithm (Dempster *et al.*, 1977), in which the expectation step of assigning each item of data to current estimates of cluster centres can be implemented in a massively parallel way (*i.e.* as many processors as there are items of data). Similarly, the maximization step of computing cluster centres by averaging can be done in batches of data assigned to each cluster. Yet, such levels of parallel hardware is not always available and at very fine grains of parallelism, communication overheads of transfering data and results between processors will become the bottleneck. Hence sequential alternatives to K-means clustering have been explored. An early approach in this direction is the *Leader-Follower* algorithm described in Duda *et al.* (2000) (Fig. 2.1). It scans through data in one-pass, associates each data with the cluster centre it is closest to, and re-estimates that cluster centre by an adaptive update. This structure has been explored in neural network settings, too, and will form the basis of the algorithms we explore in this WorkPackage.

2.2 DBSCAN:

The Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise (DBSCAN) Algorithm for clustering, introduced by Ester *et al.* (1996) (also see Daszykowski *et al.* (2001, 2002)) is a popular algorithm which aims to place cluster centres in parts of the space with high density. This algorithm sequentially scans through given data, absorbing a fraction of them as cluster centres when some pre-set conditions are met. The criteria used to include a data item as cluster center and declaring objects in its vicinity as parts of that cluster are based on the notion of *density reachable* between two points, and is driven by a threshold distance `Eps` and a minimum number of points

```
Input: $X = \{x_n^t\}_{n=1}^N, k$
Output: $C$, Idx
initialize:   $C = \left\{c_j^t\right\}_{j=1}^k$
repeat
  assign $n^{\text{th}}$ sample to nearest $c_j$
      Idx(n) $= \min_j |x_n - c_j|_2^2$
  recompute $c_j = \frac{1}{N_j}\sum_{n=j} x_n$
untill no change in $c_1, c_2, \ldots c_k$
return $C$, Idx
```

(a)

```
Input: $X = \{x_n^t\}_{n=1}^N, \theta, \eta$
Output: $C$, Idx
initialize:   $c_1 = x$
repeat
  $j \leftarrow \min_i ||x - c_i||^2$
  if $||x - c||^2 \leq \theta$
    $c_j \leftarrow c_j + \eta x$
  else
    add new $c \leftarrow x$
    normalize $c = c/||c||$
until no more patterns
```

(b)

Figure 2.1: Pseudocode of **(a)** K-Means and **(b)** *Leader-Follower* Sequential Clustering Algorithms (Duda *et al.*, 2000).

MinPoints (see pseudo-code in Fig. 2.2).

```
Input: $X, = \{x_n^t\}_{n=1}^N$, Eps, MinPoints
Output: $C$
  $\forall n \in 1:N$ do
  if $n$ not member of cluster
    Create new cluster
    while neighbours of $x_n$ satisfy
        add them to cluster
    end
  endif
  endfor
```

Figure 2.2: Pseudocode of the DBSCAN clustering algorithm (Ester *et al.*, 1996)

2.3 Resource Allocating Clustering (RAC)

The RAC algorithm, developed by Ramanan and Niranjan (2010), is a one-pass algorithm which sequentially constructs hyperspheres in the input space. A user specified parameter R is used as radius of each hypersphere. As shown in the pseudo-code in Fig, 2.3, each data item is checked to see if it falls within any of the hyperspheres placed so far. If a data does not fall into any existing hypersphere, a new one with the current data at its centre is created. While it is a simple approach, the number of clusters found by the algorithm critically depends on the parameter R. Ramanan and Niranjan (2010) show that the choice of R essentially depends on the scale of the space in which data lives and can be tuned from a subset of the data. They further show that in visual classification tasks, RAC achieves performances similar to K-Means in a fraction of the computing time. An important aspect of RAC is that it is insensitive to local density of data, whereas k-means is a density preserving algorithm. As such, the clusters achieved by RAC often retain outlier examples as cluster centres. In this case, data approximation error of RAC is higher than that of a k-means clustering solution for the same number of clusters. However, Ramanan and Niranjan (2010) argue that there are applications (such as codebook construction for image classification) in which accurate data reduction need not be the goal. Indeed, they show that the inclusion of rare data in the codebook can have small amounts of beneficial effects.

```
Input: $\boldsymbol{X}, = \{\boldsymbol{x}_n^t\}_{n=1}^N, R$
Output: $\boldsymbol{C}$
$C = \{\boldsymbol{x}_1\}$
for $n = 2 : N$
  withinSpheres = FALSE
  for $j = 1 : |\boldsymbol{C}|$
  if $||\boldsymbol{x}_n - \boldsymbol{c}_j \leq R$
    withinSpheres = TRUE
  endif
  if NOT withinSpheres
    $\boldsymbol{C} = \boldsymbol{C} \cup \boldsymbol{x}_n$
 endif
```

Figure 2.3: Pseudocode of the RAC clustering algorithm (Ramanan and Niranjan, 2010)

2.4 Sequential Input Space Carving (SISC)

A novel algorithm we developed as part of this WorkPackage combines ideas from RAC and DBSCAN. SISC is also a one-pass algorithm, which sequentially places hyperspheres of a user-specified radius in the input space. Similar to DBSCAN, it uses an additional criterion of a minimum number of points found within a hypersphere. Computational steps of the SISC algorithm are shown as pseudocode in Fig. 2.4.

We evaluated the performance of SISC on several image classification tasks by constructing codebooks of SIFT features and Support Vector Machine classifiers. Table 2.1 shows classification accuracies of several pairwise classification tasks using visual codebooks constructed by using three different clustering methods. The corresponding

```
**Input**: $\boldsymbol{X}, = \{\boldsymbol{x}_n^t\}_{n=1}^N, R, m_0$
**Output**: $\boldsymbol{C}$
Initialize: $i = 1$
repeat
  if $i = 1$
    $n \leftarrow$ random(1 : N)
  else
    $n \leftarrow \max_j ||\boldsymbol{c}_l - \boldsymbol{x}_j||^2,\ l = 1, ..., i$
  endif
 $\boldsymbol{c}_i \leftarrow \boldsymbol{x}_n$
  $\{\mathcal{D}\}_i \leftarrow \{\boldsymbol{x} \mid ||\boldsymbol{x} - \boldsymbol{c}_i||^2 \leq R\}$
  if $\mid \{\mathcal{D}\} \mid \geq m_0$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\boldsymbol{c}_i\}$
    $\mathcal{D} \leftarrow \mathcal{D} \backslash \{\mathcal{D}_i\}$
  endif
  $\{\mathcal{D}\} \leftarrow \{\mathcal{D}\} \setminus \{\boldsymbol{c}_i\}$
  $i = i + 1$
untill $\mathcal{D}$ is empty
```

Figure 2.4: Pseudocode of the SISC clustering algorithm (Mayurathan *et al.*, 2013)

computational costs are shown in Table 2.2. We see that the classification performances obtained are very similar, but achieved at a vastly reduced computational cost of the sequential algorithms. We also found that, on average, the size of the codebook constructed by SISC is smaller than that of the RAC (average across problems considered in Table 2.2 being 1388 and 610 respectively). These early results show that the new sequential algorithm is capable of finding clusters that offer competitive performance to the batch method of K-means, but at a vastly reduced computational cost.

2.4.1 Coresets

The basic idea in coresets is finding a small set of points in a lower dimension such that an $(1+\epsilon)$ approximation to the points in the original dimension is possible. Thus for a matrix $\boldsymbol{X}$ of N rows and p columns, row-wise clustering into k optimal clusters, is approximately the same as clustering the projections of the rows of $\boldsymbol{X}$ onto the first few singular vectors of $\boldsymbol{X}$. The subspace required to achieve this approximation is $\mathcal{O}\left(k/\epsilon^2\right)$ (Agarwal *et al.*, 2004; Feldman *et al.*, 2013). The use of coresets for dimensionality reduction, clustering and matrix factorization have been explored (see for example Frahling and Sohler (2008) and Har-Peled and Mazumdar (2004)).

Algorithm 2.5 shows the steps in the coreset clustering algorithm in pseudocode for-

Table 2.1: Classification performance as mean ± standard deviation on pairwise classes taken from the PASCAL VOC 2007 challenge.

Objects	K-means	RAC	SISC
Bird vs Aeroplane	81.86 ± 1.27	82.10 ± 0.80	82.39 ± 0.13
Aeroplane vs Horse	91.41 ± 2.40	87.82 ± 0.83	86.94 ± 0.72
Bicycle vs Motorbike	77.40 ± 0.04	76.10 ± 0.60	76.86 ± 0.20
Bus vs Train	79.80 ± 0.55	76.00 ± 0.60	77.53 ± 1.06
Dog vs Cat	68.22 ± 1.51	70.73 ± 0.65	71.87 ± 0.86
Cow vs Sheep	67.77 ± 1.20	71.10 ± 1.40	79.11 ± 2.04
Pot.plant vs Din.table	72.97 ± 0.53	76.00 ± 1.19	74.48 ± 0.68
Bottle vs Pottedplant	59.69 ± 2.15	67.72 ± 0.31	65.75 ± 2.12
Boat vs TV/monitor	74.85 ± 1.20	81.20 ± 1.80	82.60 ± 2.76
Aeroplane vs Boat	75.27 ± 1.80	71.20 ± 2.04	73.90 ± 0.97

Table 2.2: Computational time and size of the codebook (for 122448 SIFT descriptors) using different codebook techniques.

	Computational Time (in Sec)	Codebook size
K-means	44090	1000
RAC	82	1321
SISC	56	309

mat.

2.4.2 Experiments and Results

We evaluated the quality of clustering obtained by coreset subspace projections in visual scene recognition systems.

We used three benchmark object classification datasets to confirm the viability of coresets as a clustering method in visual codebook construction. From the `ETHZ` and `PASCAL` datasets we used all the classes, while with the `Caltech` dataset, we used a random ten class problem for the experiments reported here. We also used an upper bound of $10,000$ SIFT features (Lowe (2004)) for coreset construction.

2.5 Computation and Distortion

To evaluate if the clusterings obtained from the original 128 space and from the reduced subspaces differ, we carried out several clustering tasks with SIFT features extracted from

```
**Input**: Data : $\boldsymbol{X}, = \{\, \boldsymbol{x}_n^t \,\}_{n=1}^{N}$ , subspace dimension : $m$
**Output**: Cluster centres : $\boldsymbol{C}$

$\boldsymbol{X} \leftarrow \boldsymbol{X} - \mathrm{mean}\,(\boldsymbol{X})$
Compute first $m$ right singular vectors (> svds())
$\boldsymbol{X}_m \leftarrow \boldsymbol{U} \times \boldsymbol{D}^{(m)} \times \boldsymbol{V}^t$
Project data onto $m$ subspace: $\boldsymbol{X}_l \leftarrow \boldsymbol{X} \times \boldsymbol{V}$
Cluster $\boldsymbol{X}_l$ into $k$ clusters to give centres $\boldsymbol{C}_k^1$
Invert to original dimensions: $\boldsymbol{C}_k \leftarrow \boldsymbol{C}_k^1 \times \boldsymbol{V} + \mathrm{mean(X)}$
```

Figure 2.5: Coreset Clustering Algorithm

Dataset	Number of Classes	Number of Images
`ETHZ`	7	383
`PASCAL VOC`	20	17125
`Caltech 256`	256	30607

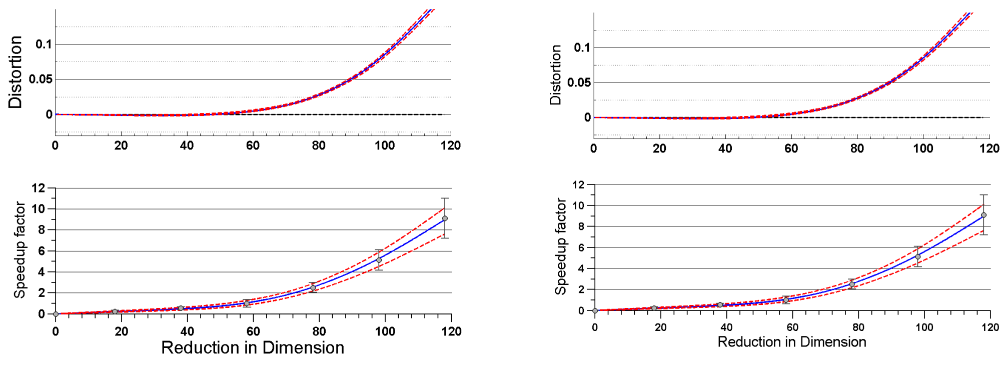
Table 2.3: Datasets used for experimental work (coreset clustering)

the various datasets used. We took $10,000$ SIFT features from random subsets of images in each class, and clustered them into approximately 150 clusters using subspaces of various dimensions. Distortions between cluster centers and the original data were meadured by summing Euclidean distances

$$d\,(\boldsymbol{X}, \boldsymbol{C}) = \sum_i \min_j \|\, \boldsymbol{x}_i - \boldsymbol{c}_j \,\|^2,$$

where, $\boldsymbol{X}$ and $\boldsymbol{C}$ denote the dataset cluster centres respectively. $\boldsymbol{x}_i$ and $\boldsymbol{c}_j$ represent individual data vectors and cluster centers, elements of $\boldsymbol{X}$ and $\boldsymbol{C}$. This is also the distortion measure used in Feldman *et al.* (2013).

Fig. 2.6 (a) and (b) show distortions and computational costs of the coreset clustering we propose, evaluated at various subspace dimensions. We plot averages (blue) and contours at one standard deviation (red) across different trials of datasets taken from the benchmark problems in Table 2.3. Clustering in the subspaces has been done with $k-$means and the RAC algorithms. The radius parameter of the RAC was set as the median of all pairwise distances between points scaled by a tuning factor. We note that this could be further tuned to potentially improve the quoted results. Computational speed is presented as a factor gained over the baseline model, which clusters in the 128 dimensional space. Looking at one point on the graphs, we can see that when the dimensionality is reduced by 100 and clustering is carried out with RAC in a 28 dimensional space, a speed up by factor six is obtainable, at the cost of less than 10% distortion.



(a) **(b)**

Figure 2.6: Distortion and gain in computational speed achieved by coreset projective clustering. (a) clustering on subspace carried out by $k-$means; and (b)clustering by RAC Ramanan and Niranjan (2010). The speed up is measured against a baseline using all 128 dimensions. Note overall computing for RAC will be much smaller than that for $k-$means.

We also verified how well the assignment of data to clusters obtained in subspace clustering agreed with clustering in the original space. To this end, we constructed a visual codebook by appending clustered data (150 visual words from each class). We then evaluated true positive and false positive rates of the original raw data finding its nearest neighbour in the group of cluster centres to which the image blonged. The hypothesis here is that the nearest cluster centre to feature vectors extracted from any image should be one of the 150 cluster centres that were derived by clustering them. Note this is done on the training data itself, and not on a set of unseen validation images.

Fig 2.7(a) and (b) show the variation in True Positive and False Positive rates as functions of computational speedup for data taken from the three datasets. Again, we show these results for two different ways of clustering on the subspace: $k-$means and RAC. We see that a range of useful gains in computational speed can be achieved with small drops in performance, measured in this manner. Note the hyperparameter in the RAC was not optimized for the subspaces, instead we used the same value throughout and there is scope to improve on this aspect. Also note that clustering performance measured this way (i.e. true and false positives of cluster assignment) is only a first check and need not directly translate into object classification performance in the subsequent phases of vector quatization of test image features and support vector machine classification (as confirmed in the next section).

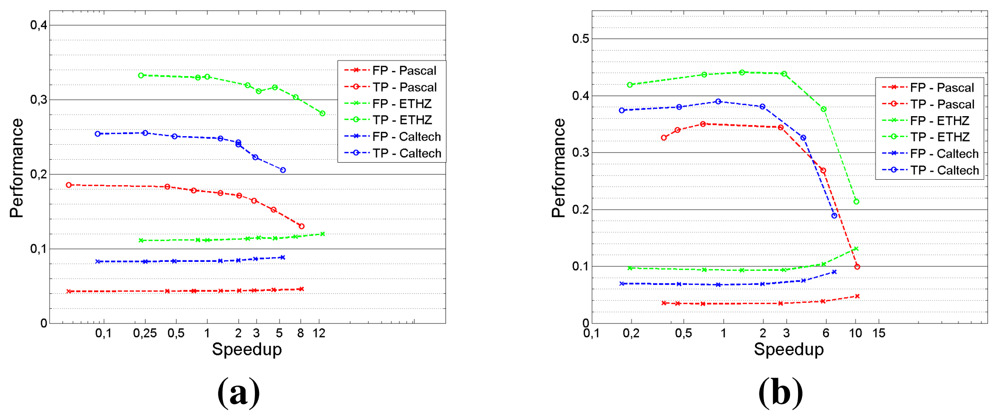


Figure 2.7: Loss of accuracy of clustering as a function of computational speed gained by coreset projective clustering. We show average true positive and false positive rates on all three of the datasets. Averages are computed over images taken from different classes. (a) clustering using k-means on subspaces, while in (b) the subspace clustering done via the RAC algorithm. Note degradation is faster in (b) because the radius parameter was not tuned in this implmentation. The speedups measured are over the baseline computational cost when clustering is done in the full 128 dimensions.

2.6 Object Recognition

We carried out object classification tasks using subsets of the three datasets and evaluated the feasibility of coreset clustering in codebook design. We implememnted $(M \times M)/2$ pairwise classifiers with $M = 7$ for the `ETHZ` dataset and $M = 10$ for the other two (`CalTech` and `PASCAL`), with randomly chosen object classes[1]. Though a significantly larger number of classifiers need to be built when compared to a one against all strategy, Milgram *et al.* Milgram *et al.* (2006) suggest this is a better approach.

Table 2.6 shows recognition accuracies (true positives) of each class in the `ETHZ` dataset when 21 binary classifiers are trained and test objects from each class is recognized. The baseline accuracies of the second column (0.433 for Apple class etc.) are comparable to previous results we have achieved on these datasets and also published by other authors (Ramanan and Niranjan, 2010). Note with seven classes a random assignment will have an accuracy of 14%.

Similarly, Tables 2.6 and 2.6 show results on the `CalTech` and `PASCAL` datasets. In all these, while there are small differences in accuracies in the recognition of the individual classes between the baseline method and the coreset clustered codebook model, the average performances are very much the same. This is being achieved at a computational saving, on average, of a factor five, across these tasks.

[1]Within the budget of the TrendMiner project, we were only able to complete such reduced experiments. Larger scale evaluations, particularly in the context of very large scale image classification task known as ImageNet are ongoing outside the scope of the project.

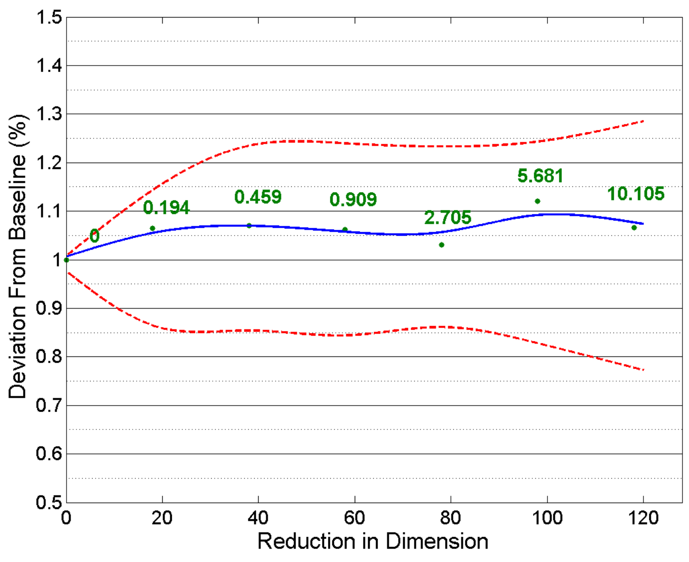


Figure 2.8: Average classification performances against baseline, computed at different subspace dimensions of codebook design. The continuous blue line shows average deviation in recognition accuracies measured against baseline accuracies, while the red dashed lines show one standard deviation away from the mean. Note, on average, performance does not deviate much from the baseline even when the dimensionality is reduced by 120, and the clustering performed on a very small subspace, giving a ten-fold reduction in computation (figures in green show speed-up obtained in the design of codebook).

2.7 Advanced Algorithms

2.7.1 Sequential Karhunen-Loeve Transform

The work carried out in this workpackage aims to use sequential algorithms as computationally efficient solutions to very large scale problems. The novel algorithm we developed, `SISC`, is capable of achieving state of the art clustering performance at a vastly reduced computational cost on image classification tasks.

However, evaluations so far have focused on large data in small dimensions (millions of SIFT features in a 128 dimensional space). Alternative problems we have not considered here exist, and are of the type in which the dimensionality can be much higher than the number of data points we have, the so called $N \ll p$ problem in statistical literature.

Such problems could not be addressed within the timescale of TrendMiner, but we are aware of solution directions. In particular, Levey and Lindenbaum (2000) suggest a sequential approach to basis function extraction, exploiting a particular shortcut to computing singular value decomposition. It is shown that the singular value decomposition of

Class	Recognition Accuracy		Computational Cost		
	Baseline	Coreset	Baseline	Coreset	Gain in Speed
Apple	0.43	0.40	349	68	5.07
Bottle	0.50	0.57	373	66	5.60
Giraffe	0.63	0.57	644	82	7.77
Hat	0.47	0.37	605	105	5.75
Mug	0.50	0.60	407	64	6.28
Starfish	0.37	0.3	508	118	4.43
Swan	0.33	0.33	317	77	4.12

Table 2.4: Recognition accuracies and computational speeds for individual classes in the `ETHZ` dataset. $k-$ means clustering was used in the reduced subspace, with dimensionality of the reduced subspace optimized.

an $M \times N$ matrix $\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{D}\,\boldsymbol{V}^t$, where $N \ll M$ can be computed efficiently via $\boldsymbol{QR}$ decomposition of the matrix and the application of SVD to a small matrix of size $N \times N$ (Golub and Van Loan, 1996). This property, coupled with incremental update of SVD of a partitioned matrix, enables Levey and Lindenbaum (2000) to derive a feature-sequential algorithm. We are confident that this can form the basis of two areas we explored in WorkPackages Two and Three in TrendMiner: (a) the bilinear model of user and tweet hubs, and (b) the Gaussian process approach with which we modelled spatial distributions of tweet sources. This integration will be interesting research to be pursued beyond the end of TrendMiner.

Class	Recognition Accuracy		Computational Cost		
	Baseline	Coreset	Baseline	Coreset	Gain in Speed
AK47	0.47	0.43	56.2	11.9	4.71
American Flag	0.53	0.60	51.4	11.3	4.55
Blimp	0.40	0.30	58.2	11.4	5.09
Floppy Disc	0.63	0.57	50.3	11.1	4.54
Guitar Pick	0.70	0.53	107.0	20.1	5.32
Hamburger	0.17	0.17	70.2	11.9	5.89
Hourglass	0.30	0.37	47.7	11.9	4.00
Iguana	0.53	0.47	89.1	14.6	6.10
Ketch-101	0.43	0.40	45.3	9.67	4.69
Knife	0.133	0.167	56.3	12.01	4.69

Table 2.5: Recognition accuracies and computational speeds for individual classes in the `CalTech` dataset.

Class	Recognition Accuracy		Computational Cost		
	Baseline	Coreset	Baseline	Coreset	Gain in Speed
Aeroplane	0.42	0.52	144	31.0	4.49
Bicycle	0.23	0.35	245	35.0	6.94
Bird	0.05	0.05	188	27.0	6.96
Boat	0.25	0.22	203	31.4	6.45
Bus	0.78	0.50	200	30.6	6.53
Car	0.33	0.23	184	35.6	5.20

Table 2.6: Recognition accuracies and computational speeds for ten individual classes in the `PASCAL` dataset.

Chapter 3

Conclusion

Concentrating on clustering, and restricting our evaluations to image classification tasks, we have explored sequential inference algorithms suitable for mining social media type data. The novel algorithm developed in this setting, Sequential Input Space Carving, is shown to be a robust way of achieving computation reduction without incurring loss of classification accuracy. Further, we have identified a particularly interesting avenue of future research to address problems in which the feature dimensionality far exceeds the amount of data items.

Bibliography

Agarwal, P., Har-Peled, S., and Varadarajan, K. (2004). Approximating extent measures of points. *Journal of the ACM*, **51**(4), 606–635.

Daszykowski, M., Walczak, B., and Massart, D. L. (2001). Looking for natural patterns in data: Part 1. density-based approach. *Chemometrics and Intelligent Laboratory Systems*, **56**(2), 83 – 92.

Daszykowski, M., Walczak, B., and Massart, D. L. (2002). Looking for natural patterns in analytical data. 2. tracing local density with optics. *Journal of Chemical Information and Computer Sciences*, **42**(3), 500–507.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, **39**(1), 1–38.

Dias, J. and Cortinhal, M. (2008). The SKM algorithm: A K-means algorithm for clustering sequential data. In H. Geffner, R. Prada, I. Machado Alexandre, and N. David, editors, *Advances in Artificial Intelligence IBERAMIA 2008*, volume 5290 of *Lecture Notes in Computer Science*, pages 173–182. Springer Berlin Heidelberg.

Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification (2nd Edition)*. Wiley-Interscience.

Ester, M., Kriegel, H.-p., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of Second International Conference on Knowledge Discovery and Data Mining, KDD-96*, pages 226–231. AAAI Press.

Farran, B., Ramanan, R., and Niranjan, M. (2009). Sequential hierarchical pattern clustering. In V. Kadirkamanathan, G. Sanguinetti, M. Girolami, M. Niranjan, and J. Noriel, editors, *Pattern Recognition in Bioinformatics*, Lecture Notes in Bioinformatics, pages 79–88. Springer.

Feldman, D., Schmidt, M., and Sohler, C. (2013). Turning big data into tiny data: Constant-size coresets for *k*-means, PCA and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1434–1453.

Frahling, G. and Sohler, C. (2008). A fast k-means implementation using coresets. *International Journal of Computational Geometry and Applications*, **18**(6), 605–625.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

Har-Peled, S. and Mazumdar, S. (2004). On coresets for K-means and K-median clustering. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 291–300, New York, NY, USA. ACM.

Kadirkamanathan, V.and Niranjan, M. and Fallside, F. (1991). Sequential adaptation of radial basis function neural networks and its application to time series prediction. In R. Lippmann, J. Moody, and D. Touretzky, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 721–727. Morgan Kauffmann.

Kadirkamanathan, V. and Niranjan, M. (1993). A function estimation approach to sequential learning with neural networks. *Neural Computation*, **5**, 954–975.

Kadirkamanathan, V., Niranjan, M., and Fallside, F. (1992). Models of dynamic complexity for time series prediction. In *Proceedings of the International Conference on Acoustics Speech and Signal Processing*, pages 269–272.

Levey, A. and Lindenbaum, M. (2000). Sequential Karhunen-Loeve basis extraction and its application to images. *Image Processing, IEEE Transactions on*, **9**(8), 1371–1374.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**(2), 91–110.

Mayurathan, B., Pinidiyaarachchi, U., and Niranjan, M. (2013). Compact codebook design for visual scene recognition by sequential input space carving. In *Machine Learning for Signal Processing (MLSP), 2013 IEEE International Workshop on*, pages 1–6.

Milgram, J., Cherier, M., and Sabourin, R. (2006). One-against-one or one-against-all: Which is better for handwriting recognition with SVMs? In *International Workshop in Frontiers in Handwriting Recognition*.

Platt, J. (1991). A resource-allocating network for function interpolation. *Neural Computation*, **3**(2), 213–225.

Ramanan, A. and Niranjan, M. (2010). A one-pass resource-allocating codebook for patch-based visual object recognition. In *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, pages 35–40.

Ramanan, A. and Niranjan, M. (2012). A review of codebook models in patch-based visual object recognition. *Journal of Signal Processing Systems*, **68**(3), 333–352.

Shadafan, R. and Niranjan, M. (1994). A dynamic neural network architecture by sequential partitioning of the input space. *Neural Computation*, **6**(6), 1202–1222.