Method to detect events in a time-series data set

Arvind Murthy
Performance Engineering Group, Yahoo!
(arvind.murthy@ymail.com)

Abstract

Time-series data is common in analytics. Data mining and analytics problems need to discover points where properties of time-series change. This generally is referred to as event detection. Some data points abnormally deviate, these are generally called outliers. Outliers also signify events. Both outliers and change-points are interesting for analytics as they suggest an unique events. Wide range of applications in analytics and data mining employ outlier and change point detection. Most previous contributions solve detection of outliers and change points separately. Method presented here is a novel unified approach for both detection of outliers and change points using a single solution. In this method, change point detection is subsumed into the problem of detecting outliers. Core idea is to compare deviation of a data point from its *short-term median* with its *long-term average*. This idea is used to construct an online algorithm that detects outliers immediately and change points, within a few time steps of their occurrence. Its worst case complexity is *O(n)*. Experiments with synthetic datasets as well as application to real datasets show why the algorithm succeeds, where others fail. This method has been currently employed to address network intrusion detection [17], video file segmentation [18] and software performance regression detection in Yahoo!.

I. Introduction

There are several solutions to discover time-points where properties of time-series change and has attracted research attention often. This problem is also referred to as event detection and encompasses a wide range of real world applications in several fields for example, in intrusion detection in computer networks[17], shot detection in video analysis systems, signal processing and performance regression detection in software systems.

From a statistical definition perspective, change point is a data point, where the underlying probability distribution of data changes significantly with respect to its parameters or even nature. e.g. the mean or variance of data changes remarkably, or the distribution changes, for example from normal to heavy tailed. An example of the latter is network traffic, which becomes heavy tailed as more and more traffic get multiplexed on the same network link.

On the other hand an outlier is a data point that deviates markedly from the rest of the data. Discriminating between outliers and change points is challenging. This is because a sudden large change in the distribution at some time point could be initially mistaken as merely an outlier with respect to the previous data points. But if these 'outliers' persist for long enough, then they are really part of a new sub-distribution.

At Yahoo! this method is being used to address the problem of change point detection. Automated performance regression detection system Trumpet [13] uses this algorithm to notify events. This system detects and alerts on changes in software performance from one build/version/release to the next. This is better than manual analysis, which is limited in accuracy and scales poorly in engineer man-hours. Currently it is being applied for alerting performance regressions in video transcoding latency. This method was used in Hadoop performance analysis. The time-series here are sequences of performance metrics of consecutive builds as measured in benchmark runs. Examples of such other metrics are transcoding throughput, run-time.
The resulting time-series of such performance metrics do not typically follow any well-known distribution. Moreover they are non-stationary i.e. their mean, variance and other statistical parameters can change over time as trends or jumps. This makes the problem very challenging.In order to adopt Continuous Integration (CI) software performance regressions need to be monitored for any integrated software stack. Performance metric data points of a software is time-series in nature. Feature additions, change in platforms and bugs affect the performance continuously. The properties of time-series keeps changing accordingly.

Current change point and outlier detection methods assume that the data follows a certain distribution and/or model the time-series as an autoregressive process. Their accuracy of detecting change points is sensitive to outliers. Some outlier detection approaches such as [1, 2, 3, 5, 10, 11] depend on static thresholds to identify outliers or have to make multiple passes, where in each pass a fixed number of outliers are detected and filtered out. Many of these methods assume a normal distribution; that makes them less robust to real world data with its arbitrary and non-stationary distributions.

Key idea in this approach is to use a ratio of deviation from short-term median to the long-term mean as a score to classify a data point. Median is taken over a shorter window prior to the current data point while the mean is taken over a longer window including the current data point. Using such sliding windows tracking trend adaptively while ignoring out-of-date statistics is achieved. This score is used to classify the current point as outlier or not. With a sequence of such outliers are detected at successive data points, retrospectively

the start of the sequence as a change point is classified. Change point detection is thus reduced to detecting sufficient number of successive outliers.

The remainder of this paper is organized as follows. In Section 2, the problem is formulated along with outlining the background. Section 3, details the proposed algorithm. A case study is presented in section 4. Experimental results on synthetic and real-world datasets is presented in Section 5. Related work is discussed in section 6.

II. Background

To illustrate the problem, consider a time-series of throughput values observed in a sorting system that sorts a fixed amount of data using different versions of software. Most of the throughput variations may be statistically regular, but once a while there may be an outlier point i.e. a marked deviation from the previous data. In general detecting such outliers is important because they may be caused by an anomaly inside the sorting software or the system environment in which it runs.

Requirements for an outlier and change detection algorithm is enumerated:
1) The process should be on-line: an outlier has to be detected as it appears. A change point has to be detected within some constant number of observations after the change happens.
2) The detection should be adaptive to a non-stationary time-series and robust to a wide variety of distributions. Preferably it should not assume specific distributions.
The major theme of this paper is, to demonstrate a unified solution for detecting outliers and change points whilst honoring the above requirements. The presented algorithm OCP_detect (Outlier Change-Point detect) honors the on-line requirement:

1) Because it computes an outlier-classification score for a data point immediately at the time at which it occurs without waiting. Change points get detected after a constant number of successive outliers happen.
2) Moreover it honors the adaptive requirement, as it uses a sliding window technique to forget old data and adapts the thresholds after change points are detected.

III. OutlierChangePoint_detect Algorithm

We denote a data sequence as $\{x_i: i = 1, 2,... N\}$, where i is time variable. p denotes the number of data points to be observed before initiating analysis. Cx_i denotes the current data point in the time series being considered for analysis. t and s denotes the thresholds for change point and outlier detection respectively. Threshold signifies the magnitude of fluctuation allowed in the data points within which they won't be classified. Window size w, v denotes the number of data points to consider in order to compute median and mean respectively. We choose median over a small window, as it is less sensitive to outliers and helps in localizing the deviation. $w<v$ in all cases. $(w/v)<0.5$ is found optimal. We maintain a vector to signify the classification state of each data point. States could be $\{0,1,2,3\}$ where '0' means neither outlier nor change point, '1' means outlier, '2' means outlier and high probability that previous point was the change point, and '3' means the previous to previous point was the change point. If the thresholds are well tuned, *OCP_detect* can detect *all* outliers and change points in a time series.

OCP_detect Algorithm (Input : p,s,t,w,v)

Step 1: Initialize $i=p+1$, iterate for all data points $\forall(i > p|p < w < v < (N - i))$

Step 2: Compute the median and mean over the windows v, w respectively.

$$\text{Median of values in window 'w'} \quad C\tilde{x}_i = \tilde{x}_{i-1}^{i-w} \tag{1}$$

$$\text{Mean of values in window 'v'} \quad C\bar{x}_i = \bar{x}_{i-1}^{i-v} \tag{2}$$

Step 3: Compute *Score1* & *Score2 (S1 and S2)*

$$Score_{1i} = \frac{|Cx_i - C\tilde{x}_i| * t}{C\bar{x}_i} \tag{3}$$

$$Score_{2i} : \frac{|C\tilde{x}_i - C\bar{x}_i|}{|C\bar{x}_i - Cx_i| * 100} \tag{4}$$

Example of a time series data points is worked out, as shown below.

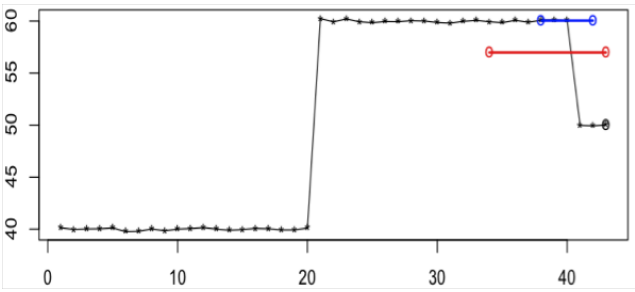

Fig. 1. Example

In the example above, Fig-1, current point is the last point in the time-series whose value is 50. The short term Median at this point is computed over the points #38 to #42 (marked blue) and the value is 60. The long term Mean for this point is computed over the points #34 to #43 (marked red) and the value is 57. The resulting scores are, Score1 : 0.880 and Score2 : 0.004 using eq.3 and eq.4 respectively.

It is observed in the above example that, mean is more sensitive to data points than median. One cannot base classification decision only considering median, however still need to quantify the deviation from mean to do this. Thus it is required to quantify the deviation relatively to mean,

median and the current data point. Considering the magnitude of deviations of current point from median, current point from mean and mean from median. Using these deviations a relation that forms the Score1 and Score2 in eq.3 and eq.4 is established.

Note that it is not possible that magnitude of deviation between mean, median and current data point be same with a distribution that does not vary over time. However they could be very close to each other.
Current data point and median being closer to each other signifies the data point is not an outlier in the short term window, this would reflect a smaller Score1. In the example above $Score1 > Score2$, i.e. current data point is away from both mean and median. Deviation of median from mean has increased compared to previous point.

Score1 relationship is the ratio of absolute difference between the current data point from the median to the mean amplified by the threshold. The median is over the short-term window *w* and mean over the longer-term window *v*. As explained using example above, taking such a ratio makes the score much more robust to fluctuations in the long-terms mean such as due to past outliers. Heuristically it is found *w/v < 0.5* to be the best choice.The resolution of detection is dependent on the threshold.

Score2 relationship is the normalized ratio of two distances (magnitudes): the median from mean and the mean from the current data point. If $Score1 > Score2$, we classify current point as an outlier. Score 2 is used as a data-dependent cutoff to classify Score1 as outlier or not. Score1 is sensitive to mean and to the deviation of current data point from median. On the other hand, Score2 is sensitive to deviation of current point from mean and to deviation of mean from median. As mean is very sensitive to fluctuations, we use the sensitivity to quantify the deviation from two reference points, the current data point and prevailing median. In window *v*, Score1 will be greater than Score2, with the presence of outliers or with data points subsequent to a change point.

Step 4: Test the Scores: *Score1 > Score2* indicates stronger possibility of current data point being an outlier or change-point. Lower and Upper limits that adapt to the trend are established, LL_i and UL_i. These serve as a cutoff reference points. These are initialized to $C\tilde{x}_i - 1\%$ and $C\tilde{x}_i + 1\%$ respectively. The limits adapt to the current median. These are help in forgetting out-of-date statistics and move ahead after a change point is encountered.

$$Score_{1i} \begin{cases} > Score_{2i} \ \wedge \\ (Cx_i < LL_i \vee Cx_i > UL_i) : V_i = 1 \\ \\ \leq S_{2i} \vee (LL_i > \ Cx_i > UL_i) : V_i = 0 \end{cases}$$

However to classify it, additional checks are made such that, the current data point is beyond a certain threshold around the median. The classification state is then saved in vector *V*. Vector *V* saves the context information of the classification decision.

Step 5: Classify outlier and change point based on the state information in vector *V*. The vector *V* helps in remembering the recent most states of previous data points. This is used to check if past outliers were actually change points.

$$V_i \begin{cases} = 1 \quad : Vs_i = (V_i + V_{i-1} + V_{i-2}) \\ = 0 \quad : Vs_i = 0 \end{cases}$$

First pass on vector Vs :

$$Vs_i{}' \begin{cases} = 3 \wedge (Vs_{i-1} = 3 \vee Vs_{i-2} = 3) \\ \ : Vs_i = 1 \\ = 1 \wedge (Vs_{i-1} = 1 \vee Vs_{i-2} = 1) \\ \ : Vs_i = Vs_i + Vs_i - 1 \end{cases}$$

Second pass on vector Vs :

$$Vs_i{}'' \begin{cases} = 0 : \text{NO change, adapt LL, UL to } C\tilde{x}_i \\ = (1 \vee 2) \wedge (Cx_i > (Cx_i + s\%Cx_i)) \\ : Outlier \\ > 2 : \text{Change point, adapt LL, UL to change} \end{cases}$$

If the current point has a higher *Score1* as indicated in V_i, past three states are considered to classify. Vector Vs_i is used to express the sum state of V_i,V_{i-1}, V_{i-2}. Vs_i stores state of the current data point with respect to past two data points. Vs_i is processed in two passes, in the first pass; if Vs_i indicates 3, it means outliers have been detected in the past and current point could be the change point. Testing the previous states it is made sure there was no change point detected in the past two data points, if detected then we infer the current point is an outlier. Similarly if the Vs_i indicates 1, it is possible that current point to be an outlier.

Finally in the second pass on Vs_i, we classify Cx_i; if the state of current point is 0, then there is no significant deviation. In case the state for current point is 1 or 2 whilst the current data point deviates more than *s%* threshold, we infer it is an outlier also state-2 signifies a higher possibility of Cx_{i-1} being a change point. If the state is 3, with accuracy we infer that two points prior to current one is the change point.

Step 6: Report the classification of outliers and change-points asin Vs_i vector.

Step 7: Persist the scores and state elements of vector *V, Vs* for past data points *N, N-1 and N-2* we compute and persist median and mean over the window sizes while classifying current data point. This step enables online implementation.

As an extension to the above outlined algorithm, upon

a new data point arrival we retrieve the state information persisted in step 7, the median, mean, and score. It is not required to process from the start of the time-series. This improves the efficiency and performance of the solution and makes the solution on-line.

IV. Case Study

Video file analysis

In order to locate a analyze video files for scene or shot detection, the inter-frame differences forms a time series data set. A frame boundary is required to be marked, where scenes change. This is the point where the frame contents change remarkably. Changes at any point in the time series during the appearance and disappearance of scenes cause abnormal points in the time series data. The underlying statistic of the time-series change. The number of changes or distance between frames t and subsequent frame $t + 1$ will be abnormal or classified as an outlier. These points signify frame boundaries. Fig-2 shows frame boundaries marked as outliers.

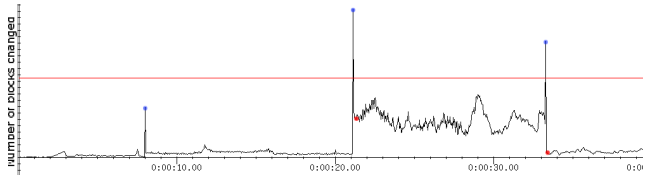

Fig. 2. Time series of frame distances over all frames of a video

Hadoop

This solution was used to detect change point in Hadoop's performance characteristics. Hadoop development results in continuous code change and feature addition by developers. The use case is to track Hadoop performance build-on-build or nightly. Change point detection reduces cost and turn around time to detect / fix performance regressions. Outliers need to be discarded as they influence further analysis. Performance of the Hadoop stack is benchmarked using a suite of tests and monitoring the metrics reported by the benchmarking system. It is evident that manual effort in achieving this does not scale.

As an example of this case study, lets us consider the sort benchmark, that sorts given amount of data on a hadoop cluster. Sort throughput is the metric that reflects the performance of the stack to sort data. The results of using this system to detect outliers and change points is shown in Fig-3

Throughput changes occur because of change in data size, change in number of nodes in the cluster or change in Hadoop software version. The change in throughput resulted in multiple change points and has been detected correctly. The solution detects all such points, marked red. Similarly, there were outliers occurring because of external factors such as network latency, node failure or choice of node that was chosen at random by Hadoop. Such points were also detected and marked blue in the above figure. Note how the algorithm adapted the LL and UL limits to the changing distribution. The analysis and notification system uses this solution and takes action upon change points.

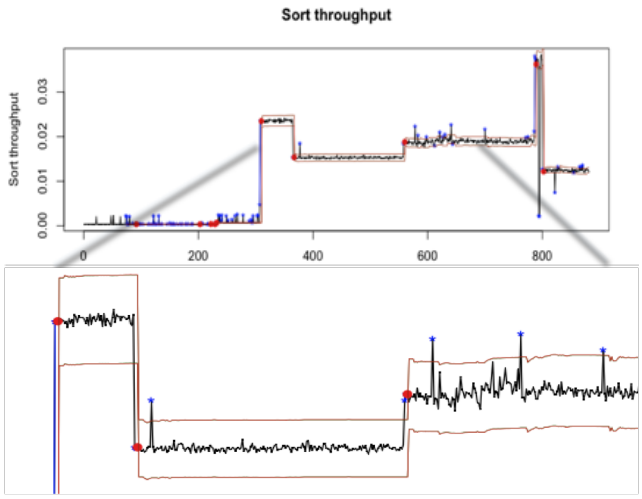

Fig. 3. Time series of sort throughput in a Hadoop cluster

V. Experiments

Experiments with synthetic data sets are performed; simulations used data generated from a mixed Gaussian distribution. For some experiments we constructed non-stationary time-series from segments of different normal distributions with standard deviation varying 1 to 2 times the mean .

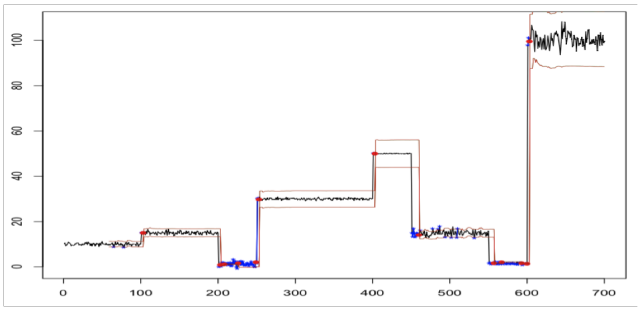

Fig. 4. Experiment data set - 1

As shown in Fig-4 shows all outliers and change points could be detected. The test data consists of small, medium and large magnitude of changes and consists outliers in each segment. The algorithm was able to detect all outliers as well as classify change points.

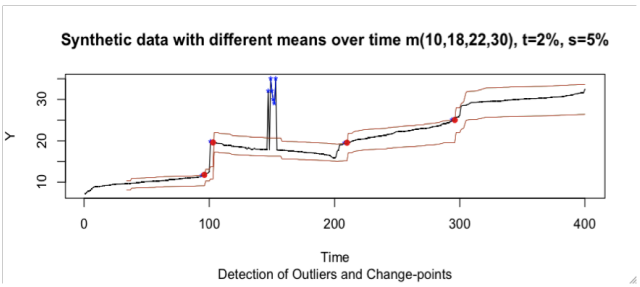

Fig. 5. Experiment data set - 2

In the second experiment referring to Fig-5, the time-series is non stationary, with a gradual trend, but in the form of very

small changes per time step. mean and median change over the sub parts. We can observe that the solution correctly detects the outliers and change points. Furthermore, the box around the time-series representing the LL and UL are adapted as per the change in mean and median over time.

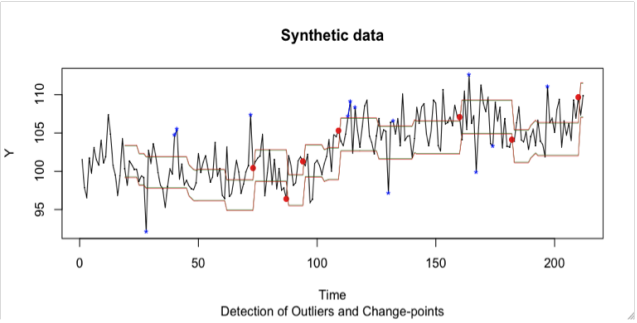

Fig. 6. Experiment data set - 3

In the third experiment, the data set has mean increasing by 6% and standard deviation range from 2 to 3. It is evident from Fig-6 that outliers are detected with s=t=2 %.

VI. Related work

Inter Quartile Range (IQR) test: IQR is based on percentile statistics. Similar to the median, percentiles presume sorting of data points in a specific manner. The sort process required could be costly, especially when the dataset is huge. For example, a data set with a billion observations to calculate the percentiles using piecewise-parabolic algorithm could take significantly more time than to generate distribution moments. Moreover detection of events are limited to threshold values chosen.

Chauvenet's criterion [10]: As stated in the criterion if the expected number of measurements at least as bad as the suspect measurement is less than 1/2, then the suspect measurement should be rejected. This cannot be applied on a non-stationary time-series data. The criteria doesn't apply, when the data distribution is strongly bi/multi-modal.

Peirce's criterion [2, 3]: As discussed in [14], theoretically accounts for the case where there is more than one suspect observation, in contrast to Chauvenet's criterion. However, Peirce's criterion can be applied more generally to a data set in Gaussian distributions.

A piecewise segmented function is employed in [9], the change points are marked as the points between successive segments. As stated in [15], a change point may be detected by discovering the points such that all errors of local model fittings of segments to the data before and after that point is minimized. However, it is computationally expensive to converge to such a point, as the local model fitting is required as many times as the number of points between the successive points every time data is input. Further, it is not guaranteed that it works well if the time-series cannot be well represented by a simple piecewise segmented function.

CUSUM (cumulative sum) [1]: Detects change in mean that are 2σ or less and success is dependent on Gaussian sequence.

Kawahara and Sugiyama [16], discuss how direct density-ratio estimation has been actively used in the machine learning community, e.g., Kernel Mean Matching [6]. However, this method is based on batch algorithm and is not suitable for change-point detection due to the sequential nature of time-series data analysis. Moreover smaller batch sizes yield inaccurate results.

We also considered Grubbs, Tietjen-Moore and Generalized Extreme Deviate Tests [11]; where these hold good for stationary time-series and assume normality of distribution.

Further, this approach is distinguished from previous works in the following regards:

1) In most previous work, outlier detection and change point detection have not been addressed together, this paper outlines the connection between them and presents a unified method for dealing with both of them from the viewpoint of non-stationary time-series. In this solution, we can detect outliers and change points simultaneously on the basis of an identical score and state vector employed in the algorithm. 2) The proposed change-detection algorithm is computationally efficient and achieves high detection accuracy. 3) In most previous work, the normality assumption is made regarding the underlying distribution, in contrast the presented solution does not and the idea is applicable to non-stationary time-series.

VII. Summary

We have proposed a solution for classifying outliers and change-points from non-stationary time series. It is addressed in two parts: scoring and classification. In the scoring we compute scores that reflect outliers, we incrementally discover and keep the state of outliers in data series. Specifically, in this solution we used sliding window to forget old statistic, thus effect of past data is gradually discounted. The algorithm is characterized in its property to address outliers and change points at the same time. This enabled us to deal with non-stationary sources. The current implementation and usage indicates the success of the algorithm.

Specifically we addressed the problem of change point detection as a problem of outlier detection. This gave a unifying view of outlier detection and change point detection.

The following issues remain for future exploration: Improve the outlier/change point detection accuracy, by adding a prediction model, thereby reduce the complexity of carefully tuning the thresholds and window sizes. Idea is to develop a model that predicts next data point with some probability.

VIII. REFERENCES

[1] Basseville, M. and Nikiforov, V., Detection of Abrupt Changes: Theory and Application, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1993.

[2] B.A. Gould, " On Peirce's criterion for the rejection of doubtful observations, with tables for facilitating its application, " Astronomical Journal IV, 83, 81 - 87 (1855).

[3] Benjamin Peirce, "Criterion for the rejection of doubtful observations, Astronomical Journal II, 45, 161- 163 (1852). (http://articles.adsabs.harvard.edu)

[4] Brodsky, B. and Darkhovsky, B., Nonparametric Methods in Change-Point Problems, Kluwer Academic Publishers, 1993.

[5] Gustafsson, F., Adaptive Filtering and Change Detection, John Wiley & Sons Inc., 2000.

[6] Huang, J., Smola, A., Gretton, A., Borgwardt, K. M. and Scholkopf, B., Correcting Sample Selection Bias by Unlabeled Data, in Advances in Neural Information Processing Systems, Vol.19, MIT Press, Cambridge, 2007.

[7] Kifer, D., Ben-David, S and Gehrke, J., Detecting Change in Data Streams, in Proc. of the 30th Int'l Conf. on Very Large Data Bases (2004), pp.180-191.

[8] V. Barnett and T. Lewis, Outliers in Statistical Data, John Wiley & Sons, 1994.

[9] V. Guralnik and J. Srivastava, Event detection from time series data, in Proc. KDD-99, pp:33-42, 1999.

[10] William Chauvenet, A Manual of Spherical and Practical Astronomy V.II , Lippincott, Philadelphia, 1st Ed (1863); Reprint of 1891 5th Ed: Dover, NY (1960).

[11] NIST / SEMATECH Statistical Methods. http://www.itl.nist.gov/div898/handbook/index.htm

[12] Apache^{TM} Hadoop^{TM}! http://hadoop.apache.org/

[13] Trumpet: Automated software stack performance analysis system. http://tiny.corp.yahoo.com/xpnV3Y-

[14] Handouts, http://classes.engineering.wustl.edu/ 2009/fall/che473/handouts/OutlierRejection.pdf

[15] Advances in Knowledge Discovery and Data Mining: 15th Pacific-Asia Conference, PAKDD 2011, Shenzhen, China, May 24-27, 2011, Proceedings

[16] Yoshinobu Kawahara and Masashi Sugiyama, Sequential Change-Point Detection Based on Direct Density-Ratio Estimation, TR09-0009 November 2009

[17]Intrusion detection, Anisha B S,Arvind Murthy, Naveen N C, IFRSA?s International Journal Of Computing, Vol2,issue 3, July 2012

[18]Video OCR, Arvind Murthy, Yahoo!, 2011