Learning from Evolving Data

Joao Gama
University of Porto, PORTUGAL

Ernestina Menasalvas
Technical Univ. of Madrid, SPAIN

Myra Spiliopoulou
University of Magdeburg, GERMANY

Athena Vakali
University of Thessaloniki, GREECE

and the Chairs of the HaCDAIS Workshop:

Mykola Pechenizkiy, Eindhoven Univ. of Technology, NETHERLANDS
Indrė Žliobaitė, Eindhoven Univ. of Technology, NETHERLANDS

UNIVERSIDADE DO PORTO

ECML-PKDD Conference
Tutorial
Barcelona, Sept. 24th, 2010

Goal for the Tutorial

Point out the new challenges introduced by evolving data like resource aware learning, change detection, novelty detection, multi-horizons analysis, and reasoning about the learning process.

- We elaborate on important application areas where data evolution must be taken into account
- we discuss the impact of evolution on economical data, and on understanding social networks;
- we investigate how learning under constraints (time, storage capacity and other resources) is affected by data evolution;
- we identify applications that require model learning over complex data (as in Customer Relationship Management or Social Tagging
- present appropriate adaptive learning methods.

Tutorial Presenters

Joao Gama

Is a researcher at LIAAD, University of Porto, working at the Machine Learning group. His main research interest is in Learning from Data Streams. He published more than 80 articles. He served as Co-chair of ECML 2005, DS09, ADMA09 and a series of Workshops on KDDS and Knowledge Discovery from Sensor Data with ACM SIGKDD. He is author of a recent book on Knowledge Discovery from Data Streams.

http://www.liaad.up.pt/~jgama/

Athena Vakali

Associate Professor at the Department of Informatics of Aristotle University, Thessaloniki, Greece. Her main research interest is Web usage mining with emphasis on social Web data and she published more than 100 papers in refereed journals and Conferences.

http://oswinds.csd.auth.gr/~avakali/

Tutorial Presenters

Myra Spiliopoulou

Professor of Information Systems in the Faculty of Computer Science, Otto-von-Guericke-Univ. Magdeburg, and Chair of the Knowledge Management & Discovery (KMD) group. Co-founder of WebKDD workshop series, has co-organized the ECML-PKDD 2006 and NLDB'08, Tutorials Co-Chair of ICDM 2010 and Workshops Chair of ICDM 2011.

http://omen.cs.uni-magdeburg.de/itikmd

Ernestina Menasalvas

Professor of DataBases at the Faculty of Computer Science, Universidad Politécnica de Madrid, Spain. Her main research interest is knowledge discovery on ubiquitous devices and resource constraint applications. She has publications in international journals and conferences on data mining processes, web mining, and techniques.

http://amaltea.ls.fi.upm.es/members/emenasalvas.htm

Tutorial Presenters (or HaCDAIS Workshop Co-chairs?)

Mykola Pechenizkiy

Assistant Professor at the Department of Computer Science, Eindhoven University of Technology, the Netherlands. He has broad research interests in data mining and its application to various (adaptive) information systems serving industry, commerse, medicine and education. He has been organizing several workshops and conferences in these areas.

http://www.win.tue.nl/~mpechen/

Indrė Žliobaitė

Postdoctoral Researcher at the Department of Computer Science, Eindhoven University of Technology, the Netherlands. She received her PhD from Vilnius University, Lithuania. Her main research interests include data mining under concept drift and context-aware prediction. She publishes in intelligent data analysis and pattern recognition venues.

http://zliobaite.googlepages.com

Tutorial Structure

Block 1: Introduction

Block 2: Supervised learning on streams

- Mining and adapting classifiers
- Dealing with concept drift
- Novelty detection

Joao Gama
Mykola Pechenizkiy
Indre Zliobaite

Block 3: Unsupervised learning on streams

- Adapting clusters
- Probabilistic models
- Learning on complex data

Myra Spiliopoulou

Tutorial Structure

Block 4: Mining evolving social data — Athena Vakali
- Structures and Models
- Community Detection in Evolving Social Graphs
- Applications of Evolving Community Detection

Block 5: Mining under resource constraints — Ernestina Menasalvas
- Introduction
- Approaches

Block 6: Conclusions and Outlook

Presentation Agenda (tentative)

☑ Block 1: Introduction

- Block 2: Supervised learning on streams (Joao Gama,Mykola Pechenizkiy,Indre Zliobaite)
- Block 3: Unsupervised learning on streams (Myra Spiliopoulou)
- Tiny Break
- Block 4: Mining evolving social data (Athena Vakali)
- Block 5: Mining under resource constraints (Ernestina Menasalvas)
- Block 6: Conclusions and Outlook

Presentation Outline

☑Block 1: Introduction

Block 2: Supervised learning on streams
- Mining and adapting classifiers (Joao Gama)
- Novelty detection (Joao Gama)
- Dealing with concept drift in AIS (M. Pechenizkiy and I. Žliobaitė)

Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Evolving Data: Illustrative Example

Wind Power Generation

At the end of 2009, the worldwide capacity of wind powered generators was 159.2 gigawatts (GW).

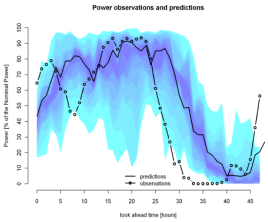
In 2020 the penetration of removable energies should be 20%

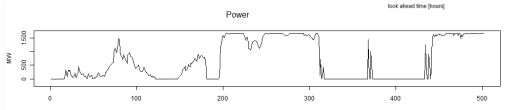
EC recommendation

Wind energy is intermittent.
Wind energy market requires predictive models

Goal:
- *Given wind velocity and direction predict the power produced by a set of turbines for a time horizon*

Predicting Wind Power



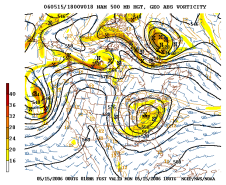


Numerical weather prediction

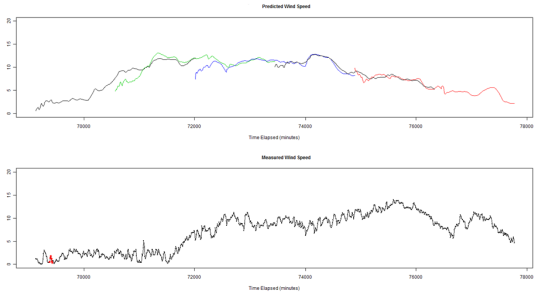
Numerical weather prediction uses current weather conditions as input into mathematical models of the atmosphere to predict the weather.

The atmosphere is a fluid. NWP uses the state of the fluid at a given tin and use the equations of fluid dynamics and thermodynamics to estimate the state of the fluid at some time in the future

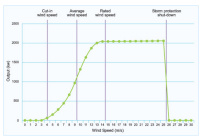
Usual NWP :
- Predictions for every hour in the next 24, 48, 72 hours
- Predictions are delivered every day



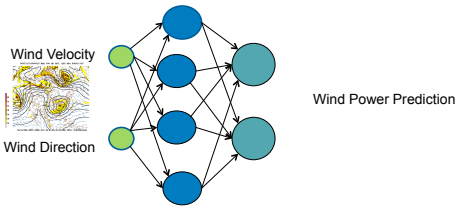
Numerical Weather Predictions



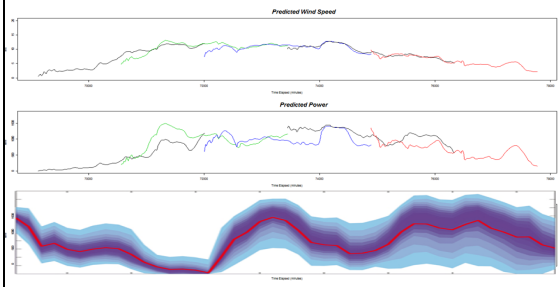
From NWP to Wind Power Prediction



From NWP to Wind Power Prediction



Wind Power Prediction



Challenges from Evolving Data

It is impractical to store and use all the historical data for training:

- it would require infinite storage and running time

Requires Incremental learning

There may be concept–drift in the data, meaning, the underlying concept of the data may change over time.

Uncertainty and reliability of predictions

Novel classes may emerge from unlabelled examples in the stream.

We might be interested in:

- prediction / forecast for different time horizon
- modeling for different time granularities
- mine evolution of the decision models based on the changes observed in a sequence of windows

Stream Classification

Construct a classification model based on past records

Use the model to predict labels for new data

- Single Classifiers
- Ensemble of Classifiers

Stream Classification

Processing each example:

- Small constant time
- Fixed amount of main memory
- Single scan of the data
- Without (or reduced) revisit old records.

Processing examples at the speed they arrive

Decision Models at anytime

Ability to detect and react to concept drift

Ideally, produce a model equivalent to the one that would be obtained by a batch data-mining algorithm

The *Very Fast Decision Tree*

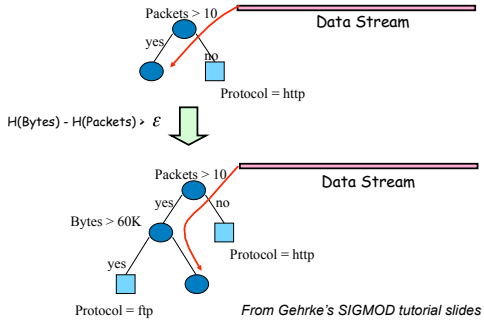
Mining High-Speed Data Streams, P. Domingos, G. Hulten; KDD00

The base Idea:

A small sample can often be enough to choose the optimal splitting attribute

- Collect sufficient statistics from a small set of examples
- Estimate the merit of each attribute
- Use Hoeffding bound to guarantee that the best attribute is really the best.
- Statistical evidence that it is better than the second best

How many examples are enough?



From Gehrke's SIGMOD tutorial slides

Hoeffding bound

Suppose we have made *n* independent observations of a random variable *r* whose range is R.

The Hoeffding bound relates the mean in the sample with the mean in the population:

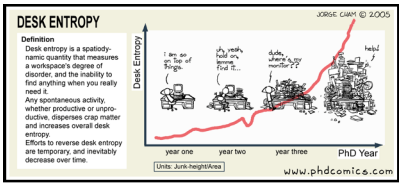
With probability 1-δ

- The true mean of *r* is in the range $\bar{r} \pm \varepsilon$ where
$$\varepsilon = \sqrt{R^2 \log(1/\delta)/(2N)}$$
- Independent of the probability distribution generating the examples.

Entropy as Splitting Criteria

How to measure the ability of an attribute to discriminate between classes?

Many measures. Entropy is a popular one: $H(x) = \Sigma\, p_i \log_2 p_i$



Growing a decision tree is guided by reducing the entropy, that is the randomness or difficulty to predict the class.

Entropy: Sufficient Statistics

Each leaf stores sufficient statistics to evaluate the splitting criterion

- For each attribute
  - If Nominal
    - Counter for each observed value per class
  - If Continuous
    - Binary tree with counters of observed values
    - Discretization: e.g. 10 bins over the range of the variable
    - Univariate Quadratic Discriminant (UFFT)

Classifying Test Examples

VFDT like algorithms can classify test examples at any time

To classify a test example

- The example traverse the tree from the root to a leaf
- It is classified using the information stored at this leaf.
  - The original VFDT classifies the test example using the majority class.

VFDT like algorithms store in leaves much more information:

- The distribution of attribute values per class.
  - Required by the splitting criteria
- Information collected from hundred's (or thousand's) of examples!

Can we use this information?

- Functional Leaves

Classification strategies

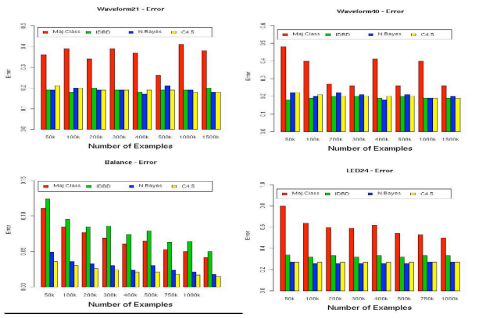
Accurate Decision Trees for mining high-speed Data Streams, J. Gama, R. Rocha; KDD03

Two classification strategies:

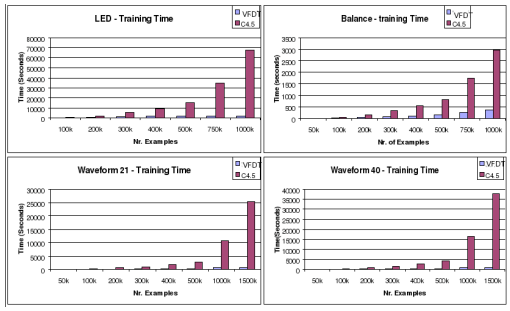
- The standard strategy use ONLY information about the class distribution: $P(y_i)$
- A more informed strategy, use the sufficient statistics $P(x_j \mid y_i)$
- Classify the example in the class that maximizes $P(y_i \mid \mathbf{X})$
  - Naive Bayes Classier: $P(y_i \mid \mathbf{X}) \propto \log(P(y_i)) + \Sigma \log (P(x_j \mid y_i))$

VFDT stores sufficient statistics of hundred of examples in the leaves.

Illustrative Evaluation



Training time / Memory used



Properties of VFDT like Algorithms

Low variance models:

- Stable decisions with statistical support.
- No need for pruning;
- Decisions with statistical support;

Low overfiting:

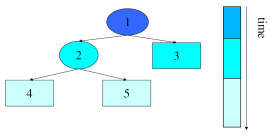
- Examples are processed only once.
- Decisions are taken using different set of examples

Convergence: VFDT becomes asymptotically close to that of a batch learner. The expected disagreement is δ/p where p is the probability that an example fall into a leaf.

Decision Processes	Batch Learning	Hoeffding Trees
Decide to expand	Choose the best split from all data in a given node	Accumulate data till there is statistical evidence in favor to a particular splitting test
Pruning	Mandatory	No need
Drift Detection	Assumes data is stationary	Smooth adaptation to the most recent concepts

VFDT like algorithms: Multi-Time-Windows

A multi-window system: each node (and leaves) receive examples from different time-windows of examples

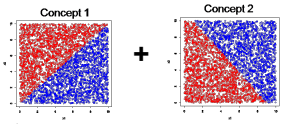


Useful for change detection:

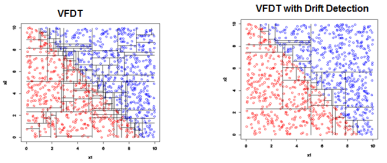
- Compare distributions at different nodes
- The RS algorithm

Automatic adaptation to change by growing the tree

The training set:



Projecting the final tree in the instance space:



References *Supervised learning*

Mining High Speed Data Streams, by P. Domingos, G. Hulten, SIGKDD 2000.

Mining time-changing data streams, G.Hulten, L. Spencer, P.Domingos, SIGKDD 2001.

Efficient Decision Tree Construction on Streaming Data, R. Jin, G. Agrawal, SIGKDD 2003.

Accurate Decision Trees for Mining High Speed Data Streams, by J. Gama, R. Rocha, P. Medas, SIGKDD 2003.

Incremental rule learning and border examples selection from numerical data streams, Ferrer-Troyano, J. Aguilar-Ruiz, and J. Riquelme (2005) Journal of Universal Computer Science 11 (8), 1426 ---1439.

J. Gratch. Sequential inductive learning.

Dimitrio Kalles. Ecient Incremental Induction of decisions trees, Machine Learning

J.Gama, R.Fernandes, and R.Rocha. Decision trees for mining data streams. Intelligent Data Analysis, 10(1):23(46, 2006.

Info-fuzzy algorithms for mining dynamic data stream, Cohen, L., G. Avrahami, M. Last, and A. Kandel (2008) Applied Soft Computing 8 (4), 1283---1294. s

Learning from time-changing data with adaptive windowing, Bifet, A. and R. Gavalda (2007). In Procs SIAM International Conference on Data Mining, pp. 443--448. SIAM.

Aggarwal, C. (Ed.) (2007). Data Streams – Models and Algorithms. Springer

J.Gama Knowledge Discovery from Data Streams, CRC Press, 2010

Ensembles of Classifiers

H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining Concept-Drifting Data Streams using Ensemble Classifiers", KDD'03.

Train **K** classifiers from **K** chunks

For each subsequent chunk

- train a new classifier
- test other classifiers against the chunk
- assign weight to each classifier
- select top **K** classifiers

Kolter and Maloof, Using additive expert ensembles to cope with Concept drift, Proc. 22° ICML, 2005

The Dynamic Weighted Majority (DWM) maintains:

- An ensemble of base learners,
- Predicts using a weighted-majority vote of these experts, and
- Dynamically creates and deletes experts in response to changes in performance.

Experts can use the same algorithm for training and prediction;

- They are created at different time steps so they use different training set of examples.
- The final prediction is obtained as a weighted vote of all the experts.
  - For each class, DWM sums the weights of all the experts predicting that class, and predicts the class with greatest weight.
- The learning element of DWM, first predicts the classification of the training example.
- The weights of all the experts that misclassified the example are decreased

Ensembles for Mining Skewed Data Streams

J.Gao, B. Ding, W. Fan, J. Han, P. Yu: Classifying Data Streams with Skewed Class Distributions and Concept Drifts. IEEE Internet Computing 12(6): 37-49 (2008)

Ubiquitous Skewed Distributions

- Examples from the interesting classes are scarce
  - Credit card frauds, network intrusions

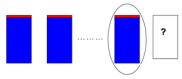
Existing Stream Classification Algorithms used to be evaluated on balanced data

Problems:

- Ignore minority examples
- The cost of misclassifying minority examples is usually huge

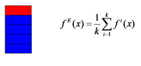
Learning from Skewed Data

Learning from batches of examples over time:
- Insufficient positive examples



STEP 1 – Sampling
- Accumulate positive examples
- Use negative examples from the last batch

STEP 2 – Generate an Ensemble
- Sample negative examples from the last batch



Analysis

Incorporation of old positive examples

- increase the training size, reduce variance
- negative examples reflect current concepts, so the increase in boundary bias is small

Ensemble

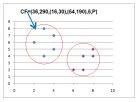
- reduce variance caused by single model
- disjoint sets of negative examples the classifiers will make uncorrelated errors

Classification on Demand

On Demand Classification of Data Streams Charu C. Aggarwal, Jiawei Han, Jianyong Wang, Philip S. Yu KDD 04

On demand classification can dynamically select the appropriate window of past training data to build the classifier.

A supervised micro-cluster for a set of:



- d-dimensional points $X_{i1} \ldots X_{in}$
- with time stamps $T_{i1} \ldots T_{in}$ and
- belonging to the class C_i

is defined as the (2d+4) tuple $(CF1^t, CF^2{}^t, CF1^x, CF^2{}^x, n, C_j)$,

- wherein $CF^2{}^x$ and $CF1^x$ each correspond to a vector of **d** entries.

Properties of micro-clusters

Each micro-cluster has a unique *id*

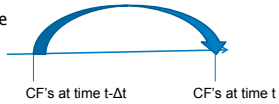
Micro clusters are additive

Incremental update of micro-cluster

- When a new data point is available:
  - Update the nearest neighbor of the same class
  - If the diameter increases to a threshold value
    - Start a new micro cluster with a single point

Micro clusters are subtractive

- $CF_t - CF_{t-\Delta t}$ summarizes the stream in the interval



Snapshots

Store the micro-clusters at different moments in time. These stored micro-cluster states are referred to as *snapshots*. Snapshots are stored in such a way that:

- Maintain sufficient amount of information about different time horizons.
- Avoid the storage of an unnecessarily large number of time horizons
- This is achieved with the use of a geometric time frame.

Frame no.	Snapshots (by clock time)
1	69 67 65
2	78 66 62
3	68 60 52
4	56 40 24
5	48 16
6	64 32

- Snapshots are classified into different frame numbers. The frame number of a particular class of snapshots defines the level of granularity in time at which the snapshots are maintained.
- Snapshots of frame number *i* are stored at clock times which are divisible by 2^i but not by 2^{i+1}.
  - » Each frame as limited capacity
- The closer to the current time, the denser are the snapshots stored.

Classification on Demand

How do we find the most effective horizon for classification at a given moment in time?

- A small portion of the training stream is not used for the creation of the micro-clusters. This portion of the training stream is referred to as the horizon fitting stream segment.
- The remaining portion of the training stream is used for the creation and maintenance of the class-specific micro-clusters

Consider an example in which:

- the current clock time is t_c, and
- a horizon of length h in order to find the micro-clusters in the time period (t_c-h, t_c).

Classification on Demand

Consider an example in which:

- the current clock time is t_c, and
- a horizon of length h in order to find the micro-clusters in the time period (t_c-h, t_c).

Finding the micro-clusters for the time period of interest

- Find the stored snapshot which occurs just before the time t_c-h.
- For each micro-cluster in the current set $S(t_c)$, we find the list of ids in each micro-cluster.
- For each *id* in the list of *ids*, we find the corresponding micro-clusters in $S(t_c-h)$, and subtract the CF vectors for the corresponding micro-clusters.
- The resulting set of micro-clusters correspond to the time horizon (t_c-h, t_c).

Classification on Demand

Evaluation

- Once the micro-clusters for a particular time horizon have been determined, they are used to determine the classification accuracy of that particular horizon.
  - This process is executed periodically in order to adjust for the changes which have occurred in the stream in recent time periods.
  - For this purpose, we use the horizon fitting stream segment.
- A data point **x** is classified in the class of the nearest neighbor micro-cluster
- Evaluate for all time horizon in the geometric time frame

Illustrative Example

Figure 5: Accuracy comparison (Synthetic dataset D10K8/C5T20, stream_speed=100, buffer_size=500, t_{gap}=25, tol_{window}=100)

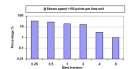


Figure 6: Distribution of the (smallest) best horizon (Synthetic dataset D100K8/C5T20, Time units=2000, buffer_size=500, t_{gap}=25, tol_{window}=100)

Predictions at different granularities

Analysis of Web click streams

- Raw data at low levels:
  - seconds, web page addresses, user IP addresses, …
- Analysts want: changes, trends, unusual patterns, at reasonable levels of details
  - E.g., *Average clicking traffic in North America on sports in the last 15 minutes is 40% higher than that in the last 24 hours."*

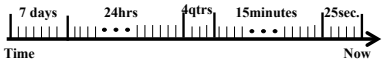
Analysis of power consumption streams

- Raw data:
  - power consumption flow for every household, every minute
- Patterns one may find: *average hourly power consumption surges up 30% for manufacturing companies in Chicago in the last 2 hours today than that of the same day a week ago*

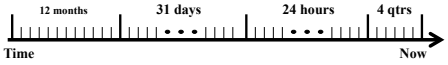
Slide from Jiawei Han

A Tilted Time-Frame Model

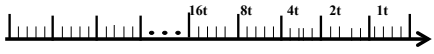
Up to 7 days:



Up to a year:



Logarithmic (exponential) scale:



Bibliography on Ensembles

Classifying Data Streams with Skewed Class Distributions and Concept Drifts, J.Gao, B. Ding, W. Fan, J. Han, P. Yu:. IEEE Internet Computing 12(6): 37-49 (2008)

Mining Concept-Drifting Data Streams using Ensemble Classifiers, H. Wang, W. Fan, P. S. Yu, and J. Han, KDD'03.

On Demand Classification of Data Streams, C. Aggarwal, J. Han, J. Wang, P. Yu; KDD 04

Online Ensemble Learning, Nikunj Oza, PhD thesis, University of California, Berkeley, 2001

Dynamic Weighted Majority: A new ensemble method for tracking concept drift, Kolter, J.Z. and Maloof, M.A., Procs Third IEEE International Conference on Data Mining, 2003

J. Kolter and M. Maloof, Using additive expert ensembles to cope with Concept drift, Proceedings of the Twenty-second International Conference on Machine Learning, 2005

Y. Yang, X. Wu and X. Zhu; Combining Proactive and Reactive Predictions for Data Streams; KDD05, 2005

N. Littlestone and M. K. Warmuth. The weighted majority algorithm. Information and Computation, 108(2):212–261, 1994.

Presentation Outline

☑Block 1: Introduction

Block 2: Supervised learning on streams

- ✓ Mining and adapting classifiers
- Novelty detection (Joao Gama)
- Dealing with concept drift in AIS

Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Novelty Detection

Classification problems where the full set of class labels is unknown.

Automatic identification of unforeseen phenomena embedded in a large amount of normal data.

Novelty is a relative concept with regard to our current knowledge:

- It must be defined in the context of a representation of our current knowledge.
- Specially useful when novel concepts represent abnormal or unexpected conditions
- Expensive to obtain abnormal examples
- Probably impossible to simulate all possible abnormal conditions

In real problems, as time goes by

- The distribution of known concepts may change
- New concepts may appear

By monitoring the data stream, emerging concepts may be discovered

Emerging concepts may represent

- An extension to a known concept (Extension)
- A novel concept (Novelty)

Several interesting applications: Early Detection of Fault in Jet Engines, Intrusion Detection in computer networks, Breaking News in a ow of text documents (news articles), Burst of Gamma-ray (astronomical data),

One-class classication

- Model knowledge about a single profile
- New examples may be identified as members of that profile or not

Methods based on Frequencies

- A pattern is surprising if the frequency of its occurrence differs substantially from that expected by chance, given the previously seen data. (TARZAN; Keogh et al., 2002)

Methods based on decision structure

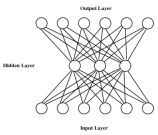
- Considers decisions taken by each unit in a decision structure.
- In a stable state, the contribution of each unit is likely to remain constant. Changes in the participation of decision units may indicate a conceptual change

One-class classification

Concept-learning in the absence of counter-examples: an autoassociaton-based approach Nathalie Japcowicz, 1999

Three layer network

- The nr. of neurons in the output layer is equal to the input layer
- The network is trained to reproduce the input at the output layer



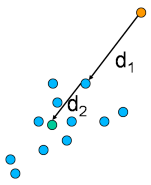
One-class classification

To classify a test example **x**

- Propagate **x** through the network and let **y** be the corresponding output;
- If $\Sigma(x_i - y_i)^2 <$ Threshold Then the example is considered from class normal;
- Otherwise, **x** is a counter-example of the normal class

Nearest Neighbor for One-class Classification

Nearest neighbour for novelty detection (Tax, 2001)



If $d_1 / d_2 > 1 \rightarrow$ reject

OLLINDA

Cluster-based novelty detection, Spinoza, Carvalho, Gama, SAC 08.

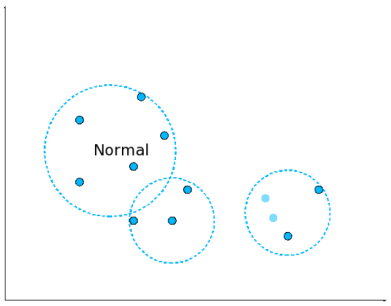
Initial Phase: Supervised, batch mode

- Start by modeling the normal condition.
- Learns a partial model about what is known.
- Based on a set of classified examples.

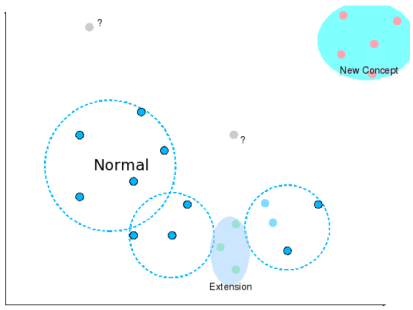
Second Phase: Process stream of unlabelled examples

- For each incoming example:
  - If it is explained by the current model: classify the example and discard
  - If it is not explained: Store in a short-term memory
  - Time to Time
    - Find clusters in the examples stored in the Short Term Memory
    - Clusters far away from existing ones: Novel concept.
    - Clusters closed to existing ones: Extend known concepts.

Initial Model



Online Phase



Bibliography on Novelty Detection

Online Novelty Detection on Temporal Sequences, by Junshui Ma, Simon Perkins, in the ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD) 2003.

A Framework for Diagnosing Changes in Evolving Data Streams, by Charu C. Aggarwal, in the ACM International Conference on Management of Data (SIGMOD) 2003.

Efficient Elastic Burst Detection in Data Streams, by Yunyue Zhu, Dennis Shasha, in the ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD) 2003.

Active Mining of Data Streams, by Wei Fan, Yi-an Huang, Haixun Wang, Philip S Yu, in the SIAM International Conference on Data Mining (SIAM DM) 2004.

OLINDDA: a cluster-based approach for detecting novelty and concept drift in data streams; E. Spinosa, A. Carvalho, J. Gama; SAC 2007: 448-452

Novelty Detection from Evolving Complex Data Streams with Time Windows, M. Ceci, Annalisa Appice, C. Loglisci, C. Caruso, F.Fumarola, D.Malerba:. ISMIS 2009: 563-572

Classification and Novel Class Detection in Data Streams with Active Mining, M. Masud, J. Gao, L. Khan, Jiawei Han, B. Thuraisingham. PAKDD (2) 2010: 311-324

Software

VFML

- http://www.cs.washington.edu/dm/vfml/
- Very Fast Machine Learning toolkit for mining high-speed data streams and very large data sets.

MOA

- http://sourceforge.net/projects/moa-datastream/
- A framework for learning from a data stream. Includes tools for evaluation and a collection of machine learning algorithms. Related to the WEKA project, also written in Java, while scaling to more demanding problems.

Rapid Miner

- http://rapid-i.com/
- The Data Stream plugin provides operators for data stream mining and for learning drifting concepts

KNIME

- http://www.knime.org/

Presentation Outline

☑Block 1: Introduction

Block 2: Supervised learning on streams

- ✓ Mining and adapting classifiers
- ✓ Novelty detection
- Dealing with concept drift in AIS

(M. Pechenizkiy and I. Žliobaitė)

Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Concept Drift: Application Perspective

CD refers to non-stationary supervised learning problems

but there are different types of CD

and different types of applications

Personal recommenders, spam filters, fraud detection, navigation are affected by drifts coming from different sources

Motivation

View CD research from an application perspective

What is the match between the mainstream CD research assumptions and properties of the applications?

Identify promising future research directions from the application perspective

We will talk about

Why changes appear in different applications?

What are the properties of CD application tasks?

How the application tasks can be categorized in terms of these basic properties?

What is Concept Drift ?

The *closed world* assumption in data mining

- learn a model from examples described by a finite set of features

In reality some important properties are not observed

- *hidden* variables that influence the concept

Hidden variables may change over time

- concepts learned at one time can become inaccurate
- possible changes in the characteristic properties of the concept

Concept Drift

- changes in the *hidden context* that can induce more or less radical changes in the *target concept*
- Virtual concept drift – changes due to population drift

Desired Properties of a System Handling Concept Drift

Adapting to concept drift asap

- must have assumptions of what and how may change

Being robust to noise and distinguishing it from concept drift

- e.g. occasionally wrong selection or rating of an item, clicking a link, connection failure (mobile computing)

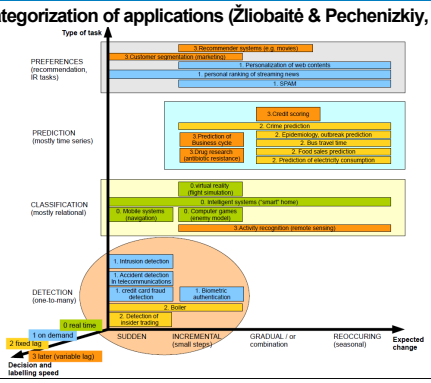
Elasticity

- discouraging brittleness

Being capable to recognize and react to reoccurring contexts

- such as seasonal differences

Categorization of applications (Žliobaitė & Pechenizkiy, 2010)



Properties of the tasks

DATA task (detection, classification, prediction, ranking)

type (time series, relational, mix)

organization (stream/batches, data re-access, missing)

DRIFT change type (sudden, gradual, incremental, reoccurring)

source (adversary, interests, population, complexity)

expectation (unpredictable, predictable, identifiable)

DECISIONS and GROUND TRUTH

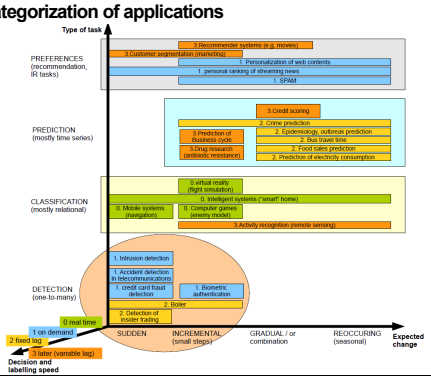
labels (real time, on demand, fixed lag, delay)

decision speed (real time, analytical)

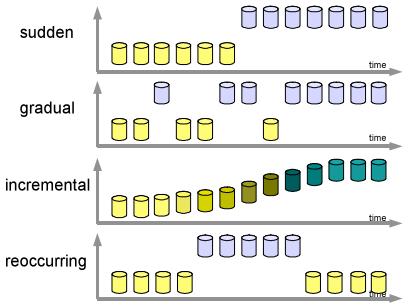
ground truth labels (soft, hard)

costs of mistakes (balanced, unbalanced)

Categorization of applications

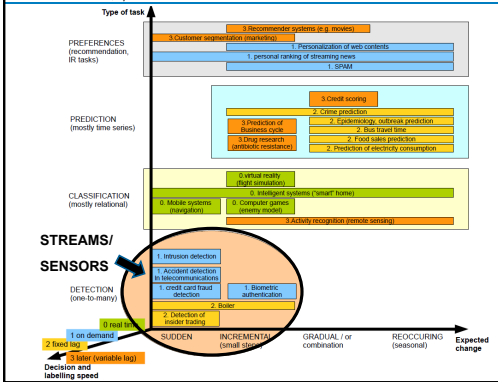


Change types

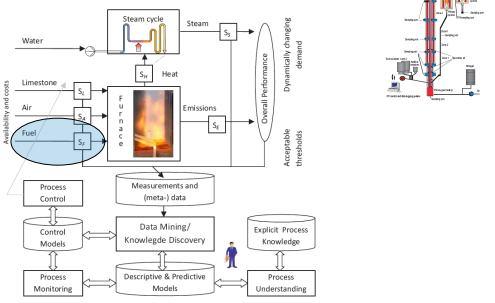


Landscape of applications

Types of apps / *Industries*	Monitoring/ control	Personal assistance/ personalization	Management and planning	Ubiquitous applications
Security, Police	Fraud detection, insider trading detection, adversary actions detection	--------	Crime volume prediction	Authentica-tion, Intrusion detection
Finance, Banking, Telecom, Credit Scoring, Insurance, Direct Marketing, Retail, Advertising, e-Commerce	Monitoring & management of customer segments, bankruptcy prediction	Product or service recommendation, including complimentary	Demand prediction, response rate prediction, budget planning	Location based services, related ads, mobile apps
Education (higher, professional, child-ren, e-Learning) Entertainment, Media	Gaming the system, Drop out prediction	Music, VOD, movie, learning object recommendation, adaptive news access, personalized search	Player-centered game design, learner-centered education	Virtual reality, simulations
...	...	...	...	...

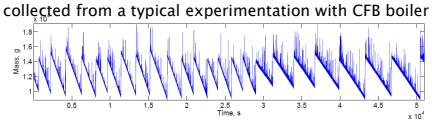


CFB Boiler Optimization



Online mass flow prediction in CFB boilers

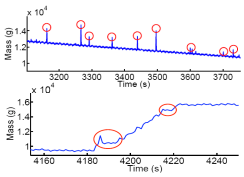
data collected from a typical experimentation with CFB boiler

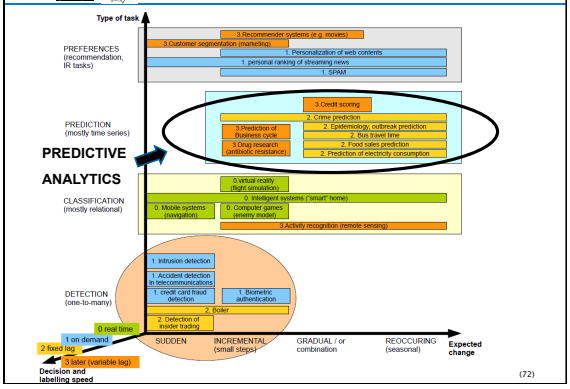


asymmetric nature of the outliers

short consumption periods within feeding stages

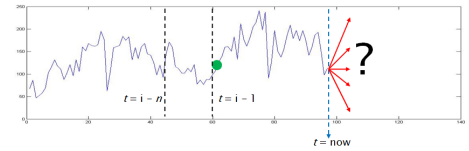
Pechenizkiy et al. 2009





Food sales prediction: utility of Belgium milk in Sep. 2009

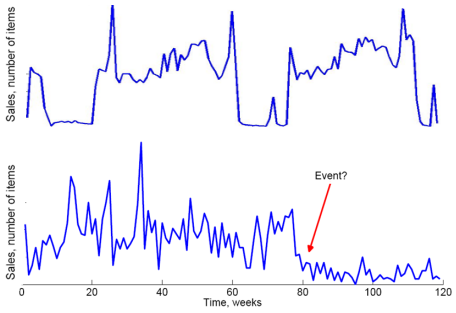
Challenges in food sales prediction (Zliobaite et al., 2009)

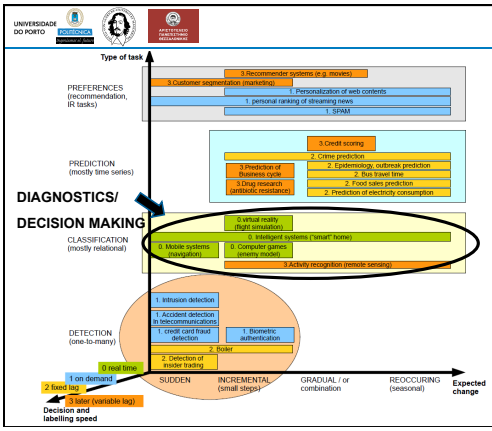


t	History	Temp	Holiday	Promo
1				
.				
i	{y(i-n) .. y(i-1)}			
.				
n				

Predict label of each new instance

Reoccuring and suddent drift in food sales



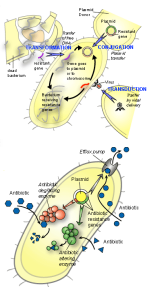


Antibiotic Resistance Prediction (Tsymbal et al., 2008)

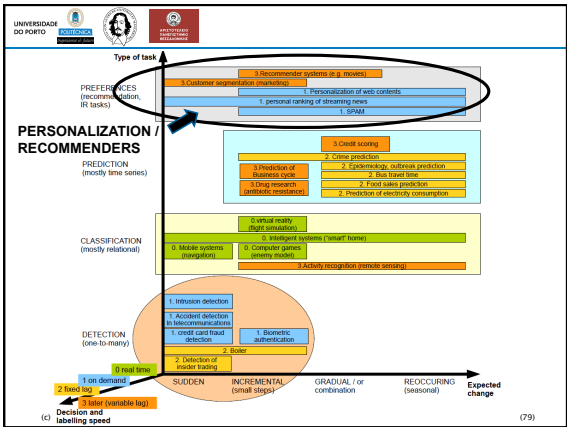
predict the sensitivity of a pathogen to an antibiotic based on data about the antibiotic, the isolated pathogen, and the demographic and clinical features of the patient.

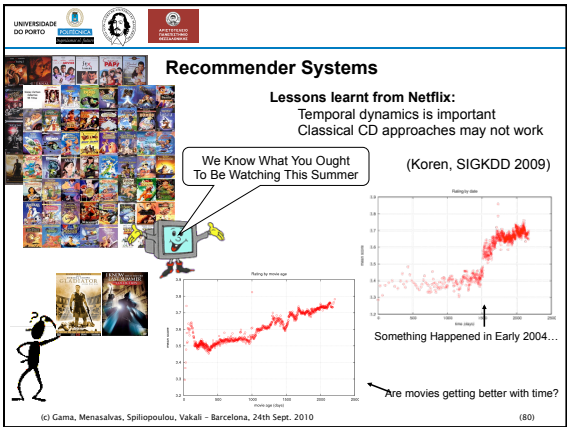
date	sex	age	isNew	days_total	days_ICU	main_dept	pathogen data	antibiotic data	sensitivity
22.1.2002	m	25	1	171	81	9	…	…	3
22.1.2002	m	25	1	171	81	9	…	…	3
22.1.2002	m	25	1	171	81	9	…	…	3
22.1.2002	m	25	1	171	81	9	…	…	3
28.1.2002	f	61	0	261	52	3	…	…	3
28.1.2002	f	61	0	261	52	3	…	…	3
28.1.2002	f	61	0	261	52	3	…	…	3
28.1.2002	f	61	0	261	52	3	…	…	1
28.1.2002	f	61	0	261	52	3	…	…	1
28.1.2002	m	25	1	171	81	9	…	…	3
28.1.2002	m	25	1	171	81	9	…	…	3
30.1.2002	m	25	1	171	81	9	…	…	3
8.2.2002	m	30	0	209	209	9	…	…	3
8.2.2002	m	30	0	209	209	9	…	…	1
8.2.2002	m	30	0	209	209	9	…	…	1
11.2.2002	f	0	0	18	0	2	…	…	1
11.2.2002	f	0	0	18	0	2	…	…	1
11.2.2002	f	0	0	18	0	2	…	…	1
new data	…	…	…	…	…	…	…	…	?
new data	…	…	…	…	…	…	…	…	?
new data	…	…	…	…	…	…	…	…	?

How Antibiotic Resistance Happens



It was on a short-cut through the hospital kitchens that Albert was first approached by a member of the Antibiotic Resistance.





Multiple sources of temporal dynamics

Both items and users are changing over time

Item–side effects:

- Product perception and popularity are constantly changing
- Seasonal patterns influence items' popularity

User–side effects:

- Customers ever redefine their taste
- Transient, short–term bias; anchoring
- Drifting rating scale
- Change of rater within household

Categorization of CD Handling Strategies

Evolving Methods vs. Informed Methods

- Evolving: adapt the learner at regular intervals without considering whether changes have really occurred
  - instance selection and instance weighting
- Informed: modify the model only after a change was detected
  - used in conjunction with a detection model

Training set manipulation vs. model manipulation

- Training set:
  - Instance weighing and selection; windowing vs. filtering
- Model manipulation:
  - VFDT and similar approaches

Outlook to Handling Concept Drift

From general methods to more specific application oriented problems like

delayed labeling

label availability

cost–benefit trade off of the model update

Changing the focus to

change description

prediction reoccurring contexts and

meta learning in addition to change detection

Current situation

Where CD research is heading to?
Where should it go to?

Come in the afternoon to HaCDAIS workshop , listen what other presenters say and share your experience and vision

References (Block 1) 2/3

Handling concept drift

Gama J., Castillo G. 2006. Learning with Local Drift Detection. ADMA 2006: 42–55

Klinkenberg R. and Renz. I. 1998. Adaptive information filtering: Learning in the presence of concept drifts. In Learning for Text Categorization, pages 33–40. AAAI Press.

Kuncheva L.I. 2008. Classifier ensembles for detecting concept change in streaming data: Overview and perspectives, Proc. 2nd Workshop SUEMA 2008 (ECAI 2008), Patras, Greece, 5–10

Kuncheva L.I. 2004. Classifier ensembles for changing environments, Proceedings 5th Int. Workshop on Multiple Classifier Systems, MCS2004, Lecture Notes in Computer Science, Vol 3077, 1–15.

Minku L. L.; White A. P.; Yao X. 2010. The Impact of Diversity on On-line Ensemble Learning in the Presence of Concept Drift., IEEE Transactions on Knowledge and Data Engineering, IEEE, v. 22, n. 5, p. 730–742.

Pechenizkiy M., Bakker J., Žliobaitė I., Ivannikov A. & Kärkkäinen T. 2009. Online Mass Flow Prediction in CFB Boilers with Explicit Detection of Sudden Concept Drift, SIGKDD Explorations 11(2).

Tsymbal A., Pechenizkiy M., Cunningham P. and Puuronen S. 2008. Dynamic Integration of Classifiers for Handling Concept Drift, Information Fusion, Special Issue on Applications of Ensemble Methods, 9(1), pp. 56-68.

Widmer G. and Kubat M. 1996. Learning in the presence of concept drift and hidden contexts. Machine Learning, 23:69-101.

Žliobaitė I., Bakker J. and Pechenizkiy M. 2009. Towards Context Aware Food Sales Prediction, In Proceedings of IEEE International Conference on Data Mining (ICDM'09) Workshops, IEEE Computer Society, pp. 94-99.

Žliobaitė I. 2010. Adaptive Training Set Formation. PhD Tesis: Vilnius University.

Žliobaitė I. 2009. Learning under Concept Drift: an overview. Technical Report: Vilnius University.

Žliobaitė I. and Pechenizkiy M. 2010. Handling Concept Drift in Adaptive Information Systems. Technical Report: Eindhoven University of Technology.

References (Block 2) 1/3

Unsupervised learning – underpinnings (3) and applications (5,6):

L. AlSumait, D. Barbara, C. Domeniconi. On-lineLDA: Adaptive topic models for mining text streams with applications to topic detection and tracking. In ICDM'08: Proc. of IEEE Int. Conf. on Data Mining, Pisa, Dec. 2008. IEEE.

H. Cao, E. Chen, J. Yang, and H. Xiong. Enhancing recommender systems under volatile user interest drifts. In CIKM'09: Proc. of the 18th ACM Conf. on Information and Knowledge Management (CIKM'09), pages 1257–1266. 2009. ACM.

Aysegul Cayci, João Bartolo Gomes, Andrea Zanda, Ernestina Menasalvas, and Santiago Eibe. Situation-Aware Data Mining Service for Ubiquitous Environments. UBICOMM '09. Malta. Oct. 2009

J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In KDD '06: Proc. of 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pages 374-383, New York, NY, USA, 2006. ACM.

Gohr, A. Hinneburg, R. Schult, M. Spiliopoulou, Topic evolution in a stream of documents. In SDM'09: Proc. of SIAM Data Mining Conf., pages 378–385, Reno, NV, USA, Apr./May 2009.

Z. Lu, D. Agarwal, and I. Dhillon. A spatio-temporal approach to collaborative filtering. In RecSys '09: Proc. of 3rd ACM Conf. on Recommender Systems, pages 13–20, New York, Nov. 2009. ACM.

Z. F. Siddiqui, M. Spiliopoulou. Combining multiple interrelated streams for incremental clustering. In SSDBM'09: Proc. of 21st Int. Conf. on Scientific and Statistical Database Management, New Orleans, USA, June 2009

Ruiz, Carlos; Menasalvas, Ernestina; Spiliopoulou, Myra. C-DenStream: Using Domain Knowledge for Clustering on Data Streams. Discovery Science '09. Porto oct. 2009

Maria, Valencia, Lauth Ina; Ernestina Menasalvas. Emerging user intentions: matching user queries with topic evolution in news text streams. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS) Vol. 17pp. 59– 80. 2009

María Valencia, Ernestina Menasalvas, and Santiago Eibe. Approach for Discovering and Tracking Local Web Search Trends. LA-WEB '09. México nov. 2009

End of Block 2

Thank you!

Questions?

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

Block 3: Unsupervised learning on streams (Myra Spiliopoulou)

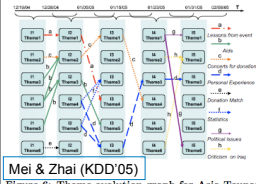
- Adapting clusters
- Probabilistic models
- Learning on complex data

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Applications of unsupervised learning on evolving data



Mei & Zhai (KDD'05)

Figure 6: Theme evolution graph for Asia Tsuna

Lin et al. (SDM'10)

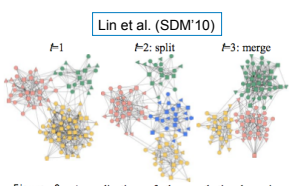


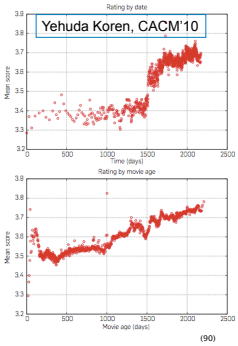
Figure 6: A realization of the synthetic dynamic tripartite networks, with number of entities N=32 (for each mode), average entity degree m=11 and p_{out}=0.05. The entities of three different modes are represented by different shapes (circle, rectangle and triangle), and different communities are indicated by different colors. There are three communities at time step t=1. At t=2, the orange community splits into two, and at t=3, the blue community merges into the green one.

Applications of unsupervised lea

- Finding hot topics in news
- Tracing changes in customer preferences for reliable prediction of demand
- Capturing community dynamics
- Detecting vessel fleets and outliers
- Making reliable recommendations

Figure 1. Two temporal effects emerging within the Netflix movie-rating dataset. *Top:* the average movie-rating made a sudden jump in early 2004 (1,500 days since the first rating in the dataset). *Bottom:* ratings tend to increase with the movie age at the time of the rating. Here, movie age is measured by the time span since its first rating event within the dataset. In both charts, each point averages 100,000 rating instances.

Yehuda Koren, CACM'10



Modeling evolving data for unsupervised learning

Data stream model of computation (Guha et al, 2003)

A *stream* is a sequence of data points $x_1, x_2, \ldots, x_i, \ldots$
that arrive in increasing order of the index i.

An algorithm operating upon a stream
is subject to memory constraints
must minimize the number of passes over the data.

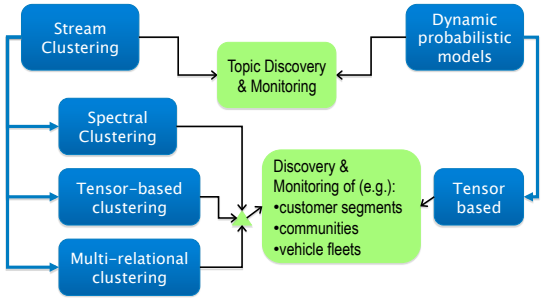
A learning algorithm operating upon a stream
must maintain a good model of the encountered data
subject to constraints on time and space.

Maintaining a good model upon *all* the data?

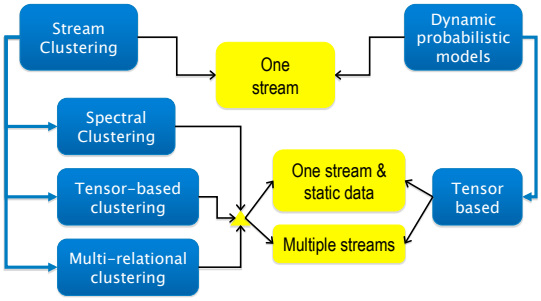
IF the data generating process is *not* stationary
THEN we want a good model on the *most recent* data.
⇒We *must forget* the oldest data.

IF the data generating process is stationary
THEN *we do not need to remember* the oldest data.
⇒We *may as well forget* the oldest data.

Unsupervised model adaptation across the time axis



Unsupervised model adaptation across the time axis



Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

Block 3: Unsupervised learning on streams (Myra Spiliopoulou)

- Adapting clusters
- Probabilistic models
- Learning on complex data

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Some core ideas in stream clustering

- Online and offline components
- Summarization of data instances
- Working with snapshots
- Reporting on cluster changes

CluStream (Aggarwal et al., 2003)

Clustering framework for evolving data streams:

- Online and offline components
  - micro-clusters
  - macro-clusters
- Summarization of the information on the stream
- Multiple time windows (snapshots) — Pyramidal time frame
- Reporting on cluster changes

CluStream (Aggarwal et al., 2003)

Let a stream of d-dimensional data points $X_1, X_2, ..., X_i, ...$ that arrive in increasing order of the index i.

Micro-cluster over

- a set of data points
- arrived at $t_{i_1}, t_{i_2}, ..., t_{i_n}$

$X_{i_1}, X_{i_2}, ..., X_{i_n}$

$$\left[\sum_{k=1}^{n} t_{i_k}\right] \quad \left[\sum_{k=1}^{n} t^2_{i_k}\right] \quad n \qquad \left[\sum_{k=1}^{n} x_{1k}\ \sum_{k=1}^{n} x_{2k}\ \ldots\ \sum_{k=1}^{n} x_{dk}\right] \quad \left[\sum_{k=1}^{n} x^2_{1k}\ \sum_{k=1}^{n} x^2_{2k}\ \ldots\ \sum_{k=1}^{n} x^2_{dk}\right]$$

CluStream (Aggarwal et al., 2003)

Let a stream of d-dimensional data points $X_1, X_2, ..., X_i, ...$ that arrive in increasing order of the index i.

Micro-clusters are stored at each *snapshot.*

Pyramidal time frame for time snapshots:

- Let α be a positive integer parameter.
- A *snapshot* of order j is taken at clock value T, where $T \bmod a^j = 0$
- retaining only the last $\alpha+1$ snapshots of order j at any timepoint.

CluStream (Aggarwal et al., 2003)

Online component for micro-cluster maintenance

For given snapshot parameter α and for number of micro-clusters q:

When enough data instances have arrived:
Build the first q clusters

When a new data instance x arrives:
IF x is close enough to the centroid of a micro-cluster M
AND IF falls within the maximum boundary of M
THEN assign it to that micro-cluster
ELSE IF x should become a new micro-cluster — Outlier
THEN either delete an old micro-cluster or merge two old ones — most proximal

At each snapshot:
Delete the data of the least recent snapshot
Compute the most recent snapshot, omitting redundant computations

CluStream (Aggarwal et al., 2003)

Offline component for macro-cluster creation:

For given time-horizon h and for number of clusters K:

Re-compute the data:
1. Find the snapshots belonging to this horizon.
2. Identify the micro-clusters to be considered
3. Re-compute the vectors of each micro-cluster, as they were within the desired snapshots

Micro-clusters have unique ids

Vectors for different datasets can be added and subtracted

Build the clusters:
1.Pick micro-cluster centroids as initial seeds with probability proportional to micro-cluster size
2.Assign each micro-cluster to its closest seed
3.Re-compute the seeds as weighted centroids

Micro-clusters are treated as data points

Report on added, deleted and retained micro-clusters

DenStream (Cao et al., 2007)

Density-based stream clustering

DBSCAN (Ester et al., 1996)
• robust to noise
• clusters need not be spheres

•online and offline components
•summarization of data instances

Micro-clusters
instead of data points

•working with snapshots

Time horizon
with an ageing function

DenStream (Cao et al., 2007)

Time horizon:

- Damped time window [0,now]
- ageing function
  that assigns exponentially decreasing weights $f(t) = 2^{-\lambda t}, \lambda > 0$

Micro-cluster:

- group of proximal data points $p_{i1}, p_{i2}, ..., p_{in}$
- that arrived at $t_{i1}, t_{i2}, ..., t_{in}$
- with joint weight $w = \sum_{j=1}^{n} f(t - t_{ij})$ at timepoint t
- with center computed as weighted average of data points
- and radius computed as weighted distance of points from center

DenStream (Cao et al., 2007)

core-micro-cluster:

- its weight is more than a threshold μ
- its radius is less than a threshold ε

potential core-micro-cluster:

- its weight is more than a threshold fragment of μ: $\beta\mu$, βin $(0,1]$
- its radius is less than ε

outlier micro-cluster:

- its weight is less than $\beta\mu$
- its radius is less than ε

DenStream (Cao et al., 2007)

Online component for micro-cluster maintenance (1/2)

For given thresholds μ, β, ε and at each new data instance x:

Identify closest p-micro-cluster
IF the *updated* radius does not exceed the threshold:
merge x to the p-micro-cluster

ELSE Identify closest o-micro-cluster
IF the *updated* radius does not exceed the threshold:
merge x to o-micro-cluster
IF the *updated* o-micro-cluster weight exceeds threshold:
promote it to p-micro-cluster

ELSE Turn x to a (singleton) o-micro-cluster

o/p-micro-clusters can be updated incrementally

DenStream (Cao et al., 2007)

Determined by the fading function

Online component for micro-cluster maintenance (2/2)

For given thresholds μ, β, ξ and every $T_{\mu,\beta}$ periods:

For each p-micro-cluster:
Recompute its weight and eventually degrade it to an o-micro-cluster

For each o-micro-cluster:
Recompute its weight
If weight is larger than threshold, promote it to a p-micro-cluster
Else If it is lower than threshold ξ, delete it.

DenStream (Cao et al., 2007)

Offline component for clustering:

DBSCAN

+

modified notion of density, using

either
- Max distance of points from micro-cluster center
- Average distance of points from micro-cluster center

ClusTree (Kranen et al., 2009)

Anytime stream clustering:

- Online and offline components
- Summarization of the information on the stream
- Working with snapshots
- Exploitation of available time for quality improvement

Anytime stream mining (Kranen et al., '09)

Principle of *anytime* algorithms:

- Deliver a model at any time
- Improve the model if more time is available

Model adaptation whenever an instance arrives

+

Model refinement whenever time permits

Slide from Kranen et al.

Anytime stream clustering with ClusTree (Kranen et al.'09)

Model:= Tree structure

node = micro-cluster

Online component:

At each data point arrival:
1.Push data point down the tree as far as time permits.
2.Take similar data points on the way down. hitchhiking
3.Grow the tree further if time permits.

Anytime stream clustering with ClusTree (Kranen et al.'09)

Activities in a node / micro-cluster:

- Insert a new data point
- Pull a data point down the subtree
- Keep a buffer of data points (hitchhikers), to be pulled down the subtree
- Aggregate data points
- Split – create subtree – for model refinement

A data point:

traverses the tree towards a leaf node,

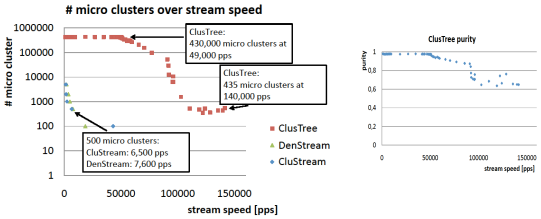
takes (max two) hitchhikers in its way down the tree,

leaves a hitchhiker at a node, if their ways do not match anymore.

Slide from Kranen et al.

Anytime stream clustering with ClusTree (Kranen et al.'09)

Number of micro-clusters and cluster purity with increasing stream speed (synthetic data):



Some core ideas in stream clustering

- Online and offline components
- Summarization of data instances
- Working with snapshots
- Reporting on cluster changes

Cluster monitoring: What is a *cluster*?

All objects of a region

- The dimensions and the metric are invariant.

All objects satisfying a function

- The dimensions are invariant.

A set of objects

MONIC (Spiliopoulou et al., 2006)

1. Partition the time axis into timepoints t_1, ..., t_m
2. Cluster (adaptively?) the data at each snapshot.
3. Compare the sets of clusters (need not be of equal strength)

Matching model:
- How to track a given cluster?
- When is a new cluster a mutation of an old one?

Transition model:
- Is an old cluster associated with exactly one new cluster?
- How did the cluster change with respect to other clusters?
- What kind of internal changes did the cluster experience?

MONIC – Matching Model (Spiliopoulou et al., 2006)

For two given clusterings ξ, ξ' (built at t_1, t_2, $t_1<t_2$) and for threshold τ:

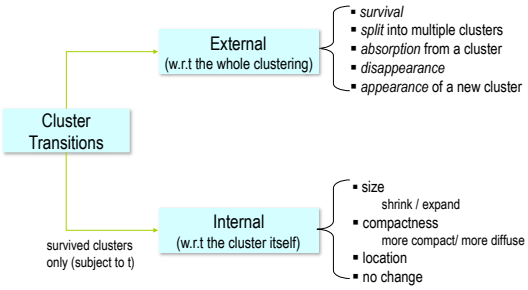
For each cluster X in ξ:

1. For each Y in ξ' built at the next timepoint t' compute

$$overlap(X,Y) = \frac{\sum_{a \in X \cap Y} age(a,t_2)}{\sum_{x \in X} age(x,t_2)}$$

2. Choose Y_{best} with max overlap to X
3. If overlap(X, Y_{best}) does not exceed τ, then discard Y_{best}

MONIC – Transition model



MONIC – Lifetime of clusters and clusterings

Lifetime of a cluster X :=

Number of adjacent timepoints where X has survived

- Strict lifetime
- Lifetime under internal transitions
- Lifetime with absorptions

Indicates the extend to which a clustering describes the data of the next timepoint.

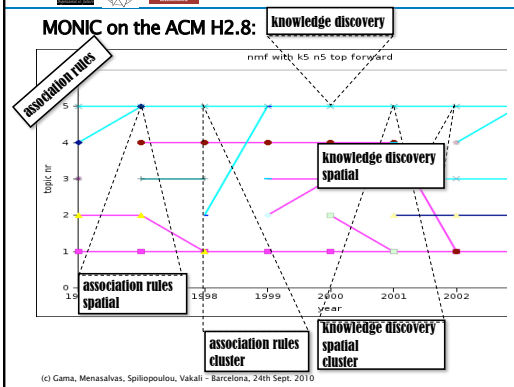
Survival ratio of a clustering :=

Portion of clusters that survive at the next timepoint

Passforward ratio of a clustering:=

Portion of clusters that survive or get absorbed at the next timepoint

MONIC on the ACM H2.8:



Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

Block 3: Unsupervised learning on streams (Myra Spiliopoulou)

- ✓ Adapting clusters
- Probabilistic models
- Learning on complex data

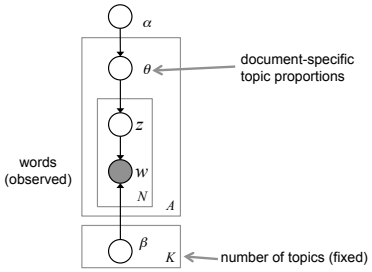
Block 4: Mining evolving social data

Block 5: Mining under resource constraints

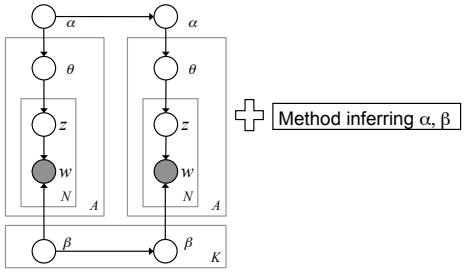
Block 6: Conclusions and Outlook

Dynamic topic model according to Blei & Lafferty (2006)

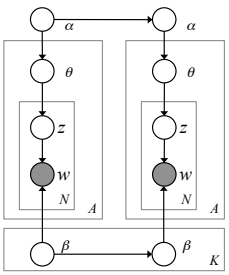
Static model with Dirichlet priors α (documents), β (topics)



Dynamic topic model according to Blei & Lafferty (2006)



Dynamic topic model according to Blei & Lafferty (2006)



k^{th} topic at slice i evolves *smoothly* from k^{th} topic of slice i-1

Learning on the whole vocabulary

Adapting a probabilistic model over a text stream

LDA thread

- L. AlSumait et al. On-line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking (ICDM'08)
- R. Nallapati et al. Multiscale Topic Tomography (KDD'07)

PLSA thread

- Qiaozhu Mei "Contextual Text Mining" – PhD thesis (2009), papers since KDD'05
- A. Gohr et al. Topic Evolution in a Stream of Documents (SDM'09)

Online LDA (AlSumait et al., 2008)

Underpinnings:

Gibbs sampling for the estimation of

- document-specific topic proportions q
- topic-specific word proportions f

Time axis is discretized – sliding window of size d slices

The model generated at slice $t-1$ is used as prior for LDA for the stream portion S^t in slice t.

Online LDA (AlSumait et al., 2008)

Underpinnings:

The model generated at slice $t-1$ is used as prior for LDA for the stream portion S^t in slice t.

- Stream S^t introduces new words.
- Parameters of topic k are determined from its past distributions:

$$\beta_k^t = \mathbf{B}_k^{t-1}\omega^\delta$$

Vector of weights for ψ, $i=\{t-d,\ldots,t\}$

Evolution matrix of topic k: has V^t columns, a column is ψ_k

V^t: number of unique words in S^t

Online LDA (AlSumait et al., 2008)

Evaluation for document modeling

- Comparison to an LDA method that remembers all data.
- Experiments on Reuters and NIPS (1988–2000)

Perplexity
towards held-out data:
lower values are better.

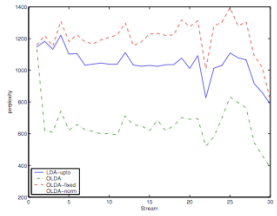


Figure 1. Comparisons of Perplexities of OLDA and LDAupto trained on Reuters

Online LDA (AlSumait et al., 2008)

Evaluation on topic evolution

- Interesting findings on the evolution of NIPS topics

Table 2. Examples of topics estimated by OLDA from NIPS corpus and its evolution over 13 years

Topic 12: Reinforcement Learning
88: state learning system states time cycles recurrence failure weight algorithm
89: node system state rule learning nodes tin match transition
90: state learning rule system node algorithm change rules controller dynamic
91: learning state reinforcement system world time adaptive planning controller
92: state learning action task exploration tasks sutton elemental
93: learning state reinforcement control time action task optimal based
94: learning state optimal control dynamic policy action time adaptive
95: learning state optimal action policy control reinforcement grid dynamic
96: learning state action policy reinforcement algorithm optimal time
97: learning state action reinforcement time policy optimal algorithm dynamic
98: state learning policy reinforcement optimal time action step control
99: state learning policy action reinforcement optimal rl time
2000: state learning policy action reward time reinforcement belief

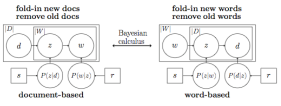
Adaptive PLSA (Gohr et al, 2009)

Underpinnings:

Time axis is discretized – sliding window of size δ slices

The model generated at slice t−1 is used as basis for PLSA adaptation for the stream portion S in slice t.

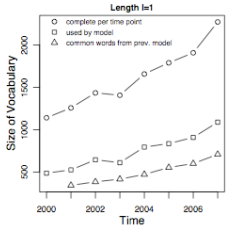
- Old documents are forgotten, new documents are folded-in
- Old words are forgotten, new words are folded-in.

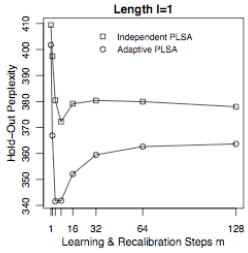


Adaptive PLSA (Gohr et al, 2009)

Evaluation for document modeling

- Comparison to a PLSA method that re-learns from scratch
- Stream of SIGIR (2000–2007)





Adaptive PLSA (Gohr et al, 2009)

Topic threads in SIGIR (2000–2007)

Thread	2000	2001	2002	2003	2004	2005	2006	2007
Evaluation	evalu search queri docu- ment relev approach re- sult method effect retriev	queri docu- ment relev evalu search retriev lan- guag result translat summari	queri retriev search term document result relev techniqu in- form translat	retriev queri system ef- fect evalu search collect inform techniqu	queri retriev relev docu- ment collect inform sys- tem feedback evalu term	queri retriev term inform feedback per- form system document evalu relev	queri retriev document result relev evalu term feedback improv precis	queri retriev relev system rank evalu perform col- lect measur effect
Presentation, later emphasis on multimedia	inform user retriev model document system base approach access studi	model re- triev inform user languag system doc- ument estim feedback problem	model in- form languag retriev user document queri col- lect method paramet	model inform retriev imag user languag video perform annot show	model retriev inform imag gang imag approach document framework distribut propos	model retriev imag languag inform approach gener propos show	model retriev inform docu- ment languag approach probabl base space framework	model re- triev inform languag base term imag propos approach show
Supervised learning	perform queri select clas- sifi method classif text vector system databas	text perform classif similar improv stori approach combin fea- tur distribut	text index classifi result sifi data corpu ap- proach time invert	text clas- sifi classifi method fea- tur result structur cat- egor term approach	featur classif text method learn algo- rithm per- form achiev automat train	method text classif featur detect extract base automat show data	text question document featur classif approach method task sentenc summar	featur learn text classif answer ap- proach paper task classifi base
Web	web qualiti page system topic answer document link automat text question task gener	web page data sys- tem analysi search page separ	web extract system queri tion answer inform data task	web re- search search page system user task result document answer evalu	web search page system user engin result algorithm inform collect rank	search web user page user inform page result network rank algorithm structur	search web search web user queri inform page engin result relev log	
Clustering	document cluster word method retriev algo- rithm inform perform sys- tem improv	document method base threshold in- dex segment score term distribut algorithm	document cluster topic algorithm set method score evalu model similar	document topic cluster method base algorithm set approach similar pro- pos	document cluster filter method al- gorithm user segment data index differ	method doc- ument topic algorithm optim semant similar data index result	cluster doc- ument data method sim- ilar semant propos align- rithm index rate	document index cluster method algo- rithm result propos op- tim perform problem

Contextual text mining with generative models (Q. Mei)

Temporal evolution of topics
over a stream D of weblog documents

Let d_t be the document arriving at timepoint t_t, i=1, 2, ...
Assume K global topics θ_1, θ_2,... θ_K and background topic θ_B.
The likelihood of word w in document d at t is:

$$p(w|d,t) = \lambda_B p(w|\theta_B) + (1-\lambda_B)\sum_{j=1}^{K} p(w,\theta_j|d,t)$$

Topic evolution on a text stream (Mei et al., 2005)

Stream of hurricane news: K=5 topics
after removal of non-discriminative words (background model), labeling of the topics and weighting the strength of the words in each topic label at each timepoint.

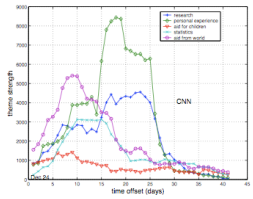


Figure 7: Absolute strength life cycles in CNN data

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

Block 3: Unsupervised learning on streams (Myra Spiliopoulou)

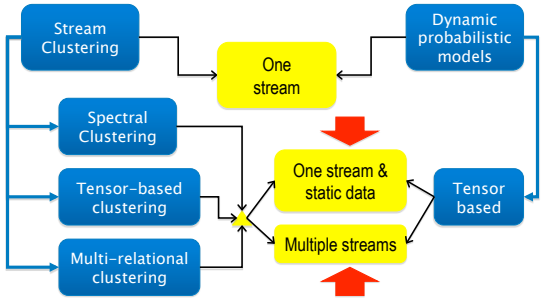
- ✓ Adapting clusters
- ✓ Probabilistic models
- ▪ Learning on complex data

Block 4: Mining evolving social data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Unsupervised model adaptation across the time axis



Example: Recommendations over users, items and tags

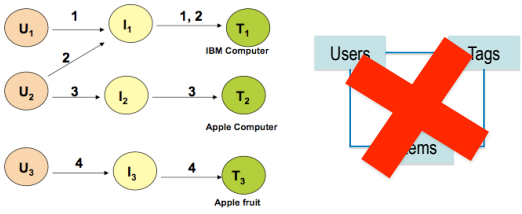


Figure 6: Usage data of the running example

P. Symeonidis, A.Nanopoulos, Y. Manolopoulos. "Tag Recommendations based on Tensor Dimensionality Reduction", Int. Conf. on Recommender Systems (RecSys'08), Lausanne, Switzerland, Oct. 2008.

Example: Recommendations over users, items and tags

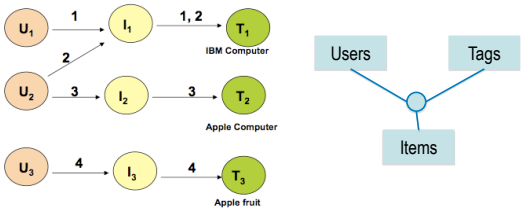


Figure 6: Usage data of the running example

P. Symeonidis, A.Nanopoulos, Y. Manolopoulos. "Tag Recommendations based on Tensor Dimensionality Reduction", Int. Conf. on Recommender Systems (RecSys'08), Lausanne, Switzerland, Oct. 2008.

Example: Recommendations over users, items and tags

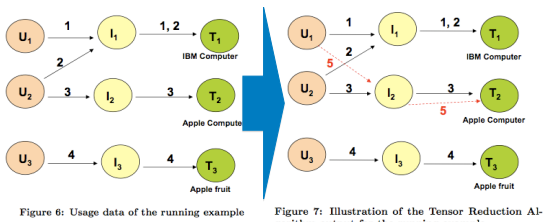


Figure 6: Usage data of the running example

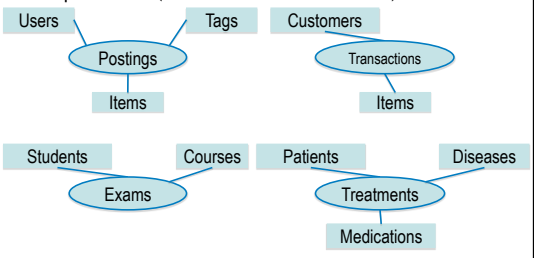
Figure 7: Illustration of the Tensor Reduction Algorithm output for the running example

P. Symeonidis, A.Nanopoulos, Y. Manolopoulos. "Tag Recommendations based on Tensor Dimensionality Reduction", Int. Conf. on Recommender Systems (RecSys'08), Lausanne, Switzerland, Oct. 2008.

Clustering over multiple, correlated streams

Challenges:

1.Multiple streams (and some *almost* static data)

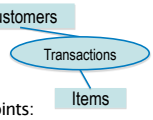


Clustering over multiple, correlated streams

Challenges:

1.Multiple streams (and some *almost* static data)

2.Correlated streams of different speeds



3.Streams of permanent objects, not of data points:
Data objects may show up more than once.

4.New objects may show up at any time.
Old objects might be forgotten (in *some* applications).

Incremental clustering + Stream clustering on complex data

How to model data and their relations?

Data on Tensor

Data on Cube (multi-relational)

- Paradigm covers graphs
- Solid theoretical underpinnings and mathematical groundwork
- Combines with advances on generative models

One topic thread towards tensor-based stream clustering

Evolutionary clustering (with temporal smoothness)

- D. Chakrabarti, R. Kumar, A. Tomkins. "Evolutionary clustering", 12th ACM SIGKDD Int. Conf (KDD'06), Philadelphia, PA, Aug. 2006

Incremental spectral clustering (with temporal smoothness)

- Y. Chi, X. Song, D. Zhou, K. Hino, B. Tseng. "Evolutionary Spectral Clustering by Incorporating Temporal Smoothness", 13th ACM SIGKDD Int. Conf. (KDD'07), San Jose, CA, Aug. 2007.

Incremental learning with generative models and temporal smoothness

- Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, B. Tseng. "FacetNet: A Framework for Analyzing Communities and their Evolutions in Dynamic Networks", World Wide Web Int. Conf. (WWW'08), Beijing, China, Apr. 2008.

Incremental tensor-based clustering

- J. Sun, D. Tao, C. Faloutsos. "Beyond streams and graphs: dynamic tensor analysis", 12th ACM SIGKDD Int. Conf. (KDD'06), Philadelphia, Aug. 2006
- Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, A. Kelliher. "MetaFac: Community Discovery via Relational Hypergraph Factorization", ACM SIGKDD Int. Conf. (KDD'09), Paris, France, June-July 2009

Evolutionary clustering (Chakrabarti et al., '06), (Chi et al., '07)

1. Two aspects of model quality (Chakrabarti et al.,'06):
   - Snapshot cost CS — captures quality of current clustering
   - Temporal cost CT — captures similarity to previous clustering
2. Model learning as optimization problem for

$$Cost(\xi) = a \times CS(\xi) + \beta \times CT(\xi)$$

- Find a sequence of models that minimizes cost (Chakrabarti et al.,'06) — k-means, agglomerative hierarchical
- Build a model that minimizes cost wrt previous model (Chi et al.,'07) — k-means, spectral for graph partitioning

Modeling temporal smoothness (Chi et al.,'07)

Snapshot cost of clustering ξ' at t':

$$CS(\xi',t') := SSE(\xi',t') = \sum_{j=1}^{K} \sum_{x \in C_{j,t'}} \left\| \frac{\sum_{y \in C_{j,t'}} y}{|C_{j,t'}|} - x \right\|^2$$

k-means

Two functions for temporal cost of clustering ξ' at t' towards clustering ξ at t (t:=t'-1):

1) Preserving Cluster Quality (PCQ):

SSE of the elements in ξ' *as they were at t*
towards
their non-normalized centers *at t*,
normalized by the cardinalities of the clusters *at t'*

NOT ORIGINAL NOTATION !

Modeling temporal smoothness (Chi et al.,'07)

Snapshot cost of clustering ξ' at t':

$$CS(\xi',t') := SSE(\xi',t') = \sum_{j=1}^{K} \sum_{x \in C_{j,t'}} \left\| \frac{\sum_{y \in C_{j,t'}} y}{|C_{j,t'}|} - x \right\|^2$$

k-means

Two functions for temporal cost of clustering ξ' at t' towards clustering ξ at t (t:=t'−1):

2) Preserving Cluster Membership (PCM):

$$CT_{PCM}(\xi',\xi) = -\sum_{X \in \xi'} \sum_{Y \in \xi} \frac{|X \cap Y|^2}{|X| \cdot |Y|}$$

originally assuming that |ξ|=|ξ'|=K, and relaxing this later on.

NOT ORIGINAL NOTATION !

Modeling temporal smoothness (Chi et al.,'07)

Snapshot cost of clustering ξ' at t':

$$CS(\xi',t') := SSE(\xi',t') = \sum_{j=1}^{K} \sum_{x \in C_{j,t'}} \left\| \frac{\sum_{y \in C_{j,t'}} y}{|C_{j,t'}|} - x \right\|^2$$

k-means

Two functions for temporal cost of clustering ξ' at t' towards clustering ξ at t (t:=t'−1):

2) Preserving Cluster Membership (PCM):

$$CT_{PCM}(\xi',\xi) = -\sum_{X \in \xi'} \sum_{Y \in \xi} \frac{|X \cap Y|^2}{|X| \cdot |Y|}$$

originally assuming that |ξ|=|ξ'|=K, and relaxing this later on.

NOT ORIGINAL NOTATION !

Symmetric function; contrast to MONIC's asymmetric best-matching model

One topic thread towards tensor–based stream clustering

Evolutionary clustering (with temporal smoothness)

- D. Chakrabarti, R. Kumar, A. Tomkins. "Evolutionary clustering", 12th ACM SIGKDD Int. Conf (KDD'06), Philadelphia, PA, Aug. 2006

Incremental spectral clustering (with temporal smoothness)

- Y. Chi, X. Song, D. Zhou, K. Hino, B. Tseng. "Evolutionary Spectral Clustering by Incorporating Temporal Smoothness", 13th ACM SIGKDD Int. Conf. (KDD'07), San Jose, CA, Aug. 2007.

Incremental learning with generative models and temporal smoothness

- Y.–R. Lin, Y. Chi, S. Zhu, H. Sundaram, B. Tseng. "FacetNet: A Framework for Analyzing Communities and their Evolutions in Dynamic Networks", World Wide Web Int. Conf. (WWW'08), Beijing, China, Apr. 2008.

Incremental tensor–based clustering

- J. Sun, D. Tao, C. Faloutsos. "Beyond streams and graphs: dynamic tensor analysis", 12th ACM SIGKDD Int. Conf. (KDD'06), Philadelphia, Aug. 2006
- Y.–R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, A. Kelliher. "MetaFac: Community Discovery via Relational Hypergraph Factorization", ACM SIGKDD Int. Conf. (KDD'09), Paris, France, June–July 2009

MetaFac (Lin et al., KDD'09)

1. Community–related data modeled on multiple tensors
   - *Facet*: set of objects of the same type (tensor mode)
   - *Relation:* interaction among facets – at least binary
   - *Multi–relational hypergraph*: facets & relations among them
2. Metagraph factorization for community discovery
3. Generalization for evolving data:
   - Given a metagraph G and a timestamped sequence of data tensors defined on G, find a non–negative core tensor and corresponding factors/facets for each timepoint t
   - assuming consistent interactions in a community

More on tensor-based clustering for community discovery in Block 4

Incremental clustering + Stream clustering on complex data

How to model data and their relations?

Data on Tensor	Data on Cube (multi–relational)
+ Paradigm covers graphs + Solid theoretical underpinnings and mathematical groundwork + Combines with advances on generative models	+ Paradigm covers categorical data + Covers re-appearing, fading and new objects + Combines with existing stream algorithms
– Does not cover categorical data – All objects to come must be known in advance (heuristics may be used instead)	– Scales well to number of modes and badly to number of dimensions – Memory management is a challenge

Mining a multi–table data *set*

Naive solution:

- Join the tables and
- mine the result

Violates the core assumption of tuple independence

Running Example

Transaction

TID	CID	PID	Time
1	1	1	1
2	1	1	1
3	2	1	1
4	2	1	1
5	1	3	2
6	1	3	2
7	2	1	2
8	1	3	3
9	3	3	3
10	4	4	4

Product

PID	Name	Price	CatID
1	Web Analysis	100	1
2	Insomnia	50	2
3	25th Hour	25	2
4	Judgement	30	3

Customer

CID	Name	Status	Income
1	David	S	20000
2	Tom	M	65000
3	Harris	M	90000
4	John	M	90000

Category

Name	CatID
Book	1
DVD	2
CD	3
Shoe	4

Fig. 2: Schema of the multi-table stream. As the Transaction stream is faster and contains tuples that may be forgotten after a certain period of time, it is assigned a time-based sliding window of 2 units. The other three streams i.e. Customer, Product and category contains object that must be remembered, therefore each of them is assigned a cache and a secondary for long term maintenance of their tuples. Each of them has a fixed cache size of two tuples.

Mining a multi-table data set (Kroegel, 2003)

Non-naive solution: ***Propositionalization***

- Specify a target table T
- For each tuple x in T
  and tupleset T_x in a table T'
  - build four columns per numerical attribute in T'
    and fill in the min, max, avg and count of values for x in T_x
  - build one column per value of nominal attribute in T' and fill it

  as extensions to T
- Mine the extended table T

Propositionalization in the example dataset

Transaction

TID	CID	PID	Time
1	1	1	1
2	1	1	1
3	2	1	1
4	2	2	1
5	1	3	2
6	1	3	2
7	2	1	2
8	1	3	3
9	3	3	3
10	4	4	4
11	4	4	4

Product

PID	Name	Price	CalD
1	Web Analysis	100	1
2	Insomnia	50	2
3	25th Hour	25	2
4	Judgement	38	3

Customer

CID	Name	Status	Income
1	David	S	20000
2	Tom	M	65000
3	Harris	M	90000
4	John	M	90000

Category

Name	CalD
Book	1
DVD	2
CD	3
Shoe	4

CID	Name	Married	Income	Count	AVG_P	MIN_P	MAX_P	Count_Book	Count_DVD	Time
1	David	S	20000	2	100	100	100	2	0	1
2	Tom	M	65000	2	75	50	100	1	1	1
1	David	S	20000	4	62.5	25	100	2	2	2
2	Tom	M	65000	3	83.3	50	100	2	1	2
1	David	S	20000	3	25	25	25	0	3	3
3	Harris	M	90000	1	25	25	25	0	1	3
4	John	M	90000	2	38	38	38	2	0	4

Mining a multi-table data stream

Challenges:

- Dealing with the past

  Which data to forget ...
  ... for each table?
  Which attribute values to forget?
- Dealing with the future

  How many nominal values may show up per attribute?
  How long to wait until a tuple shows up?

Mining a multi-table stream (Siddiqui et al., 2009)

Components:

- Typification of streams with respect to oblivion — window-based / cache-based
- Tuple ranking — **Stream preparation at timepoint t_i**
  Management of windows and caches
- Tuple expansion with data from multiple streams
- Incremental clustering algorithm (Siddiqui & Spiliopoulou, DS'09)
- Incremental tree induction algorithm (__, SSDBM'10)

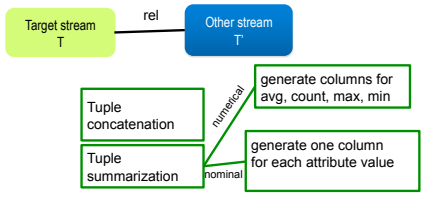
Preprocessing at each mining run: Tuple ranking

For each visible tuple x of target stream T:

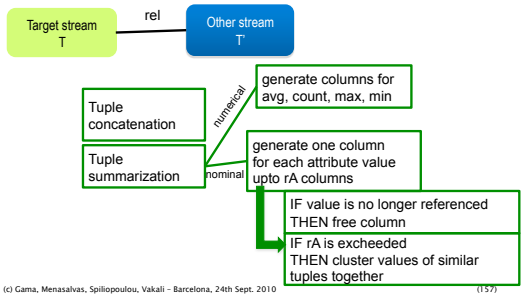
- For each stream T' associated with window
  expand x with the tuples inside the window
- For each stream T' associated with a cache
  - weight each tuple in T' with the number of references to it
  - assign a bonus to tuples that are already in the cache
  - apply a decay function to reduce the weight of older tuples
  - update the cache with the top-rank tuples,
    fetching them from secondary storage if necessary

Simplified

Preprocessing: Tuple expansion



Preprocessing: Tuple expansion



Mining a multi-table stream (Siddiqui et al., 2009)

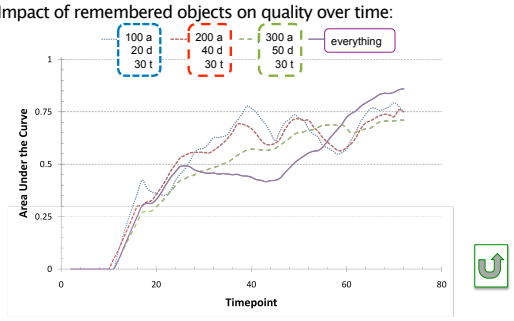
Components:

- ✓ Typification of streams with respect to oblivion — window-based, cache-based
- ✓ Tuple ranking — Stream preparation at timepoint t_i
- ✓ Management of windows and caches
- ✓ Tuple expansion with data from multiple streams

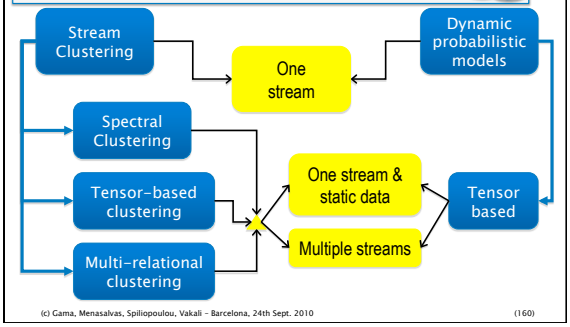
- Incremental clustering algorithm (Siddiqui & Spiliopoulou, DS'09)
- Incremental tree induction algorithm (__, SSDBM'10)

Tree Induction on PKDD'99 competition data (Siddiqui et al., 10)

Impact of remembered objects on quality over time:



New book "SOCIAL NETWORK DATA ANALYTICS", ed. Charu Aggarwal:
Chapter 6 "EVOLUTION IN SOCIAL NETWORKS: A SURVEY"
by Myra Spiliopoulou — to appear



References (Block 3) (1/3)

C. Aggarwal, J. Han, J. Wang, P. Yu. "A Framework for Clustering Evolving Data Streams", 29th Int. Conf. on Very Large Data Bases (VLDB'03), Berlin, Germany, 2003.

F. Cao, M. Ester, W. Qian, A. Zhou. "Density-Based Clustering Over an Evolving Data Stream with Noise", SIAM Int. Conf. on Data Mining (SDM'06), 2006.

S. Guha, A. Meyerson, N. Mishra, R. Motwani, L. O' Callaghan. *Clustering data streams: Theory and practice*. IEEE Trans. of Knowledge and Data Eng., 15(3):515-528, 2003.

P. Kranen, I. Assent, C. Baldauf, T. Seidl. "Self-Adaptive Anytime Stream Clustering", IEEE Int. Conf. on Data Mining (ICDM'09), Miami, Florida, Dec. 2009

M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, R. Schult. "MONIC – Modeling and Monitoring Cluster Transitions", 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'06), Philadelphia, PA, Aug. 2009

References (Block 3) (2/3)

L. AlSumait, D. Barbara, and C. Domeniconi. On-line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking. 2008 IEEE Conf. on Data Mining (ICDM'08), 373-382, Pisa, Italy, Dec. 2008.

D. Blei, J. Lafferty. Dynamic topic models. 2006 Int. Conf. on Machine Learning (ICML'06), 2006.

A. Gohr, A. Hinneburg, R. Schult, M. Spiliopoulou. Topic evolution in a stream of documents. 2009 SIAM Data Mining Conf. (SDM'09), Reno, CA, Apr.-May 2009.

Q. Mei, C. Zhai. Discovering evolutionary theme patterns from text – an exploration of temporal text mining. 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'05), 198-207, Chicago, IL, Aug. 2005.

Q. Mei. *Contextual Text Mining*. PhD thesis.University of Illinois at Urbana-Champaiign, 2009

References (Block 3) (3/3)

Y. Chi, X. Song, D. Zhou, K. Hino, B. Tseng. "Evolutionary Spectral Clustering by Incorporating Temporal Smoothness", ACM SIGKDD Int. Conf. (KDD'07), San Jose, CA, Aug. 2007.

Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, B. Tseng. "FacetNet: A Framework for Analyzing Communities and their Evolutions in Dynamic Networks", World Wide Web Int. Conf. (WWW'08), Beijing, China, Apr. 2008.

Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, A. Kelliher. "MetaFac: Community Discovery via Relational Hypergraph Factorization", ACM SIGKDD Int. Conf. (KDD'09), Paris, France, June-July 2009.

Z. Siddiqui, M. Spiliopoulou. "Combining Multiple Interrelated Streams for Incremental Clustering", 21st Int. Conf. on Scientific and Statistical Database Management (SSDBM09), New Orleans, May 2009

Z. Siddiqui, Myra Spiliopoulou. "Clustering a Stream of Growing Objects", Discovery Science Int. Conf. (DS'09), Porto, Oct. 2009

References (Block 3) (further citations)

G. Hulten, L. Spencer, P. Domingos. "Mining Time-Changing Data Streams", 7th ACM SIGKDD Int. Conf. (KDD '01), 97-106, ACM, New York, NY, USA, 2001

Mark A. Kroegel. On Propositionalization for Knowledge Discovery in Relational Databases. PhD thesis, University of Magdeburg, Germany, 2003.

R. Nallapati et al. "Multiscale Topic Tomography", ACM SIGKDD Int. Conf. (KDD'07), San Jose, CA, Aug. 2007

Ning et al. "Incremental spectral clustering by efficiently updating the eigen-system" in Pattern Recognition 43(1), 113-127, 2010

Z. Siddiqui, M. Spiliopoulou. "Tree Induction over Perennial Objects", In Proc. of 22nd Int. Conf. on Scientific and Statistical Database Management (SSDBM'10), LNCS 6187, 640-657, Heidelberg, June-July 2010

P. Symeonidis, A.Nanopoulos, Y. Manolopoulos. "Tag Recommendations based on Tensor Dimensionality Reduction", Int. Conf. on Recommender Systems (RecSys'08), Lausanne, Switzerland, Oct. 2008. D. Chakrabarti, R. Kumar, A. Tomkins. "Evolutionary clustering", 12th ACM SIGKDD Int. Conf (KDD'06), Philadelphia, PA, Aug. 2006

J. Sun, D. Tao, C. Faloutsos. "Beyond streams and graphs: dynamic tensor analysis", 12th ACM SIGKDD Int. Conf. (KDD'06), Philadelphia, PA, Aug. 2006

End of Block 3

Thank you!

Questions?

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data (Athena Vakali)

- Structure and Models
- Community Detection in Evolving Social Graphs
- Applications of Evolving Community Detection

Block 5: Mining under resource constraints

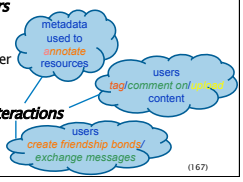
Block 6: Conclusions and Outlook

Introduction

- The Web (and especially Web 2.0) has become a source of vast amounts of social data which are evolving in fast rates.
- Users participate massively in Web 2.0 applications such as:
  - social networking sites (e.g. facebook),
  - blogs, microblogs (e.g. Mashable, twitter),
  - social bookmarking/tagging systems (e.g. del.icio.us, flickr)
- Social data refer to different types of ***actors***
  - users
  - content
  - metadata

  handled separately or together

  that are associated via various types of ***interactions***



Motivation for Mining Social Data

- The availability of massive sizes of data gave new impetus to data mining.
  - e.g. more than **400 million active Facebook users**, sharing on average more than **25 billion pieces of content** each month [Facebook Statistics 2010]
- Mining social web data can act as a barometer of the users' opinion. Non-obvious results may emerge.
  - Collaboration and contribution of many individuals formation of ***collective intelligence***
  - ***Wisdom of the crowd***: more accurate, unbiased source of information.
- Social data mining results can be useful for applications such as recommender systems, automatic event detectors, etc
- Various mining techniques are/can be used: **community detection**, clustering, statistical analysis, classification, association rules mining, …

[Facebook Statistics 2010] http://www.facebook.com/press/info.php?statistics

Static vs Evolving social data

Assume data from a social networking website spanning the time period of the year 2009…

Static social data

User u_1 has commented on user's u_2 posts 20 times in 2009.

User u_1 has three friends, users u_2, u_4, and u_{10}.

Evolving social data

User u_1 has commented on user's u_2 posts 10 times in February 2009, 10 times in April 2009 and 0 times during the rest of 2009.

User u_1 has only one friend until May 2009, user u_2. In June 2009 she makes two new friends, users u_4 and u_{10}.

Time granularity

Which information is richer?

Motivation for Mining Evolving Social Data

- Analysis of social data referring to a given time period is of *great importance*. However, as social data are evolving rapidly, the analysis of the evolution of social data over time is deemed **crucial**.

Applications:

- Identifying over time **the events** that affect social interactions
  - ✓ tracking posts in a micro-blogging website to identify floods, fires, riots, or other events and inform the public
- Highlighting **trends** in users' opinions, preferences, etc
  - ✓ companies can track customers' opinions and complaints in a timely fashion to make strategic decisions
- Tracking the **evolution** of groups (*communities*) of users or resources, finding changes in time and correlations
  - ✓ Develop better personalized recommender systems to improve user experience
  - ✓ scientists can more easily identify and relate social phenomena

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data (Athena Vakali)

- Structure and Models
- Community Detection in Evolving Social Graphs
- Applications of Evolving Community Detection

Block 5: Mining under resource constraints

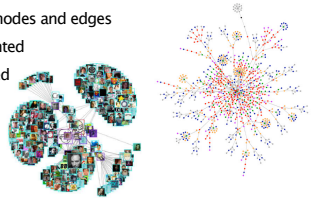
Block 6: Conclusions and Outlook

Structures and models

The network model as an obvious choice…

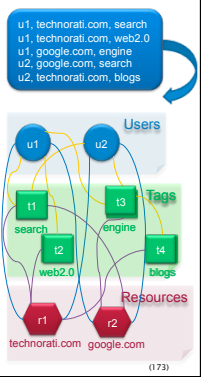
- Social data are interconnected through associations forming a **network** or **graph** $G(V, E)$, where V is the set of nodes and E is the set of edges.
  - ***nodes*** represent entities/objects and ***edges*** represent relations
  - different types of nodes and edges
  - weighted/unweighted
  - directed/undirected

In a weighted network each edge (i,j) has an arithmetic value (weight) $w_{i,j}$ denoting its significance



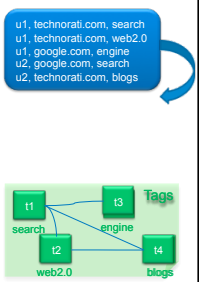
Structures for static social data

- **Hypergraph**: generalization of a graph where an edge connects more than one nodes
- **Folksonomy**: lightweight knowledge representation emerging from the use of a shared vocabulary to characterize resources – *tripartite hypergraph* [Hotho06, Mika05]
- Projection on *bipartite* & *unipartite graphs* for simplicity [Au Yeung09]
  - e.g. tag–tag network where two tags are connected if they have been assigned to the same resource
- Graph's structure can be encoded in an **adjacency matrix** ***A*** if G: unweighted ($A[i,j] = 1$, $(i,j) \in E$) or a **similarity matrix** ***M*** if G: weighted ($M[i,j] = w_{i,j}$ (i,j), $(i,j) \in E$)

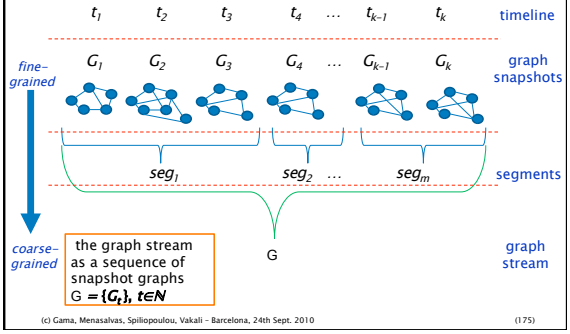


Structures for static social data

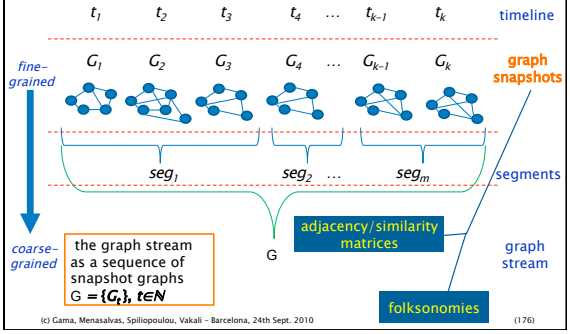
- **Hypergraph**: generalization of a graph where an edge connects more than one nodes
- **Folksonomy**: lightweight knowledge representation emerging from the use of a shared vocabulary to characterize resources – *tripartite hypergraph* [Hotho06, Mika05]
- Projection on *bipartite* & *unipartite graphs* for simplicity [Au Yeung09]
  - e.g. tag–tag network where two tags are connected if they have been assigned to the same resource
- Graph's structure can be encoded in an **adjacency matrix** ***A*** if G: unweighted ($A[i,j] = 1$, $(i,j) \in E$) or a **similarity matrix** ***M*** if G: weighted ($M[i,j] = w_{i,j}$ (i,j), $(i,j) \in E$)



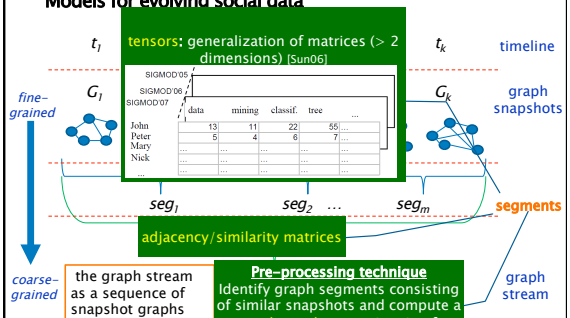
Models for evolving social data



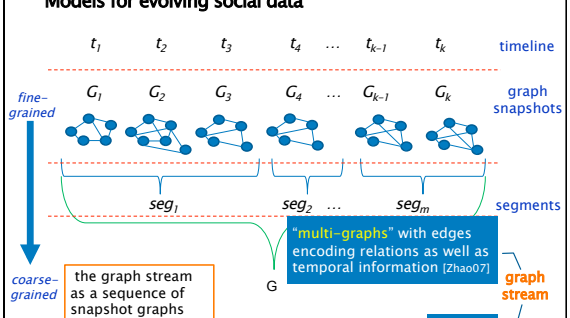
Models for evolving social data



Models for evolving social data



Models for evolving social data



Assumptions for modeling graph updates

Evolving data demand evolving structures. **But how can a graph structure change over time?**

Assuming discrete time-steps

Between time-step ***t*** and ***t+1*** :

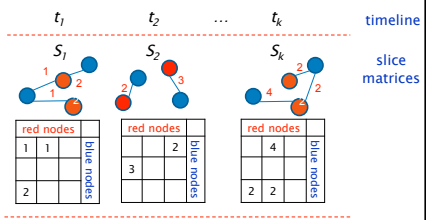
- only one update operation can take place (*single update*)
- multiple update operations can take place (*batch update*)

Evolution of graph structure G_t to G_{t+1}

- **Growth** – arrival of new nodes
- **Shrinkage** – departure of existing nodes
- **Densification** – creation of new edges,
- **Sparsification** – deletion of existing edges
- **Weight updates** – weight increase / decrease for existing edges

Evolution of the graph model using aggregation techniques

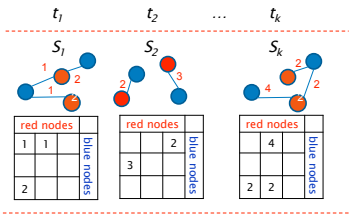
Evolving bipartite graphs as sequences of observed new edges and weights. Updates are modeled with **slice matrices** $S_1, S_2, \ldots, S_k$, $k \in N$ [Tong08]



- The graph at time-slice t_k can be created by **aggregating** the updates observed at previous time-slices.
- Creation and update of a "time aggregate adjacency matrix" *M(t)* placing different emphasis on links based on their age:
  (i) Global Aggregation, (ii) Sliding Window, (iii) Exponential Weighting

Evolution of the graph model using aggregation techniques

Evolving bipartite graphs as sequences of observed new edges and weights. Updates are modeled with **slice matrices** $S_1, S_2 \ldots, S_k$, $k \in \mathbb{N}$ [Tong08]



E.g. if $k = 3$, applying the **global aggregation** scheme:

$M(t_1) = \begin{bmatrix} 1 & 1 & \\ & & \\ 2 & & \end{bmatrix}$ $M(t_2) = \begin{bmatrix} 1 & 1 & 2 \\ 3 & & \\ 2 & & \end{bmatrix}$ $M(t_3) = \begin{bmatrix} 1 & 5 & 2 \\ 3 & & \\ 4 & 2 & \end{bmatrix}$

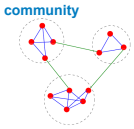
Presentation Outline

- ☑Block 1: Introduction
- ☑Block 2: Supervised learning on streams
- ☑Block 3: Unsupervised learning on streams
- Block 4: Mining evolving social data (Athena Vakali)
  - ✓ Structure and Models
  - ▪ Community Detection in Evolving Social Graphs
  - ▪ Applications of Evolving Community Detection
- Block 5: Mining under resource constraints
- Block 6: Conclusions and Outlook

Communities in social data

Definitions

- *Groups of vertices which probably share common properties and/or play similar roles in the graph.* [Fortunato07]
- *Sets of nodes more densely connected to each other than to the rest of the graph vertices.* [Girvan02]



Explicit communities are easily identified, e.g. groups in Facebook

Implicit communities are hidden and (graph) analysis is required for their identification, e.g. Flickr clusters

Community detection algorithms aim to identify implicitly-defined communities in graphs with graph mining techniques.

Numerous communities in Web 2.0 social networks:
- **tags** describing the same concept (tag clouds)
- **users** sharing common interests
- or **combined** communities of both users and tags

Motivation for community detection in social data

- Interesting to look for communities since they form functional units (e.g. sets of resources relevant to a topic) for the *local* (within group) and the *global* (over the whole graph) level.
- Community detection can influence tasks such as:
  - ✓ designing crawl strategies on the Web
  - ✓ predicting evolution of network-based data collected from the Web
  - ✓ developing improved methodologies for content outsourcing, recommendation, etc
  - ✓ revealing (hidden) emerging phenomena on the Web
  - ✓ understanding social interactions in Web 2.0 settings
  - ✓ automatic event identification from social data

Community Detection in Social Graphs

Measures

Quantitative, for assessing relations between nodes	Qualitative, for evaluating community structure
co-occurrence	modularity
cosine similarity	local modularity
tf- idf	cut-size
betweeness centrality	node outwardness
structural similarity	

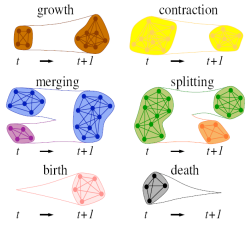
Extra challenges: overlapping of hierarchical communities

Methodologies

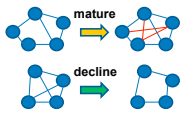
Graph partitioning	Spectral algorithms
Clustering	Methods based on statistical inference
Divisive algorithms	Dynamic algorithms
Modularity-based methods	

Community Evolution

- Community structure changes as the underlying graph evolves
- **Community evolution patterns**



[Palla07]



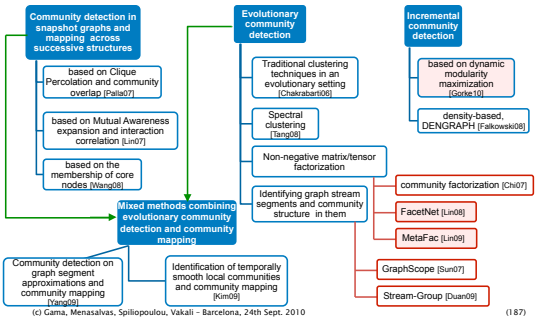
[Falkowski06]

Recent research areas

- Identification of community evolution
- Evolutionary community detection
- Incremental community detection

Community Detection in Evolving Social Graphs

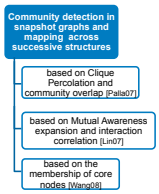
Classification of methods for community detection in evolving graphs

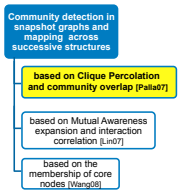


Community detection in snapshot graphs and mapping of communities across successive community structures

Typical procedure:

1. Static snapshot graphs for each time-step
2. Traditional community detection technique applied on each snapshot
3. For each snapshot: mapping of each community to its predecessor and successor using a similarity measure and/or temporal smoothing technique

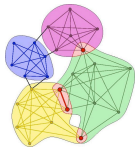




Community Detection with Clique Percolation Method (CPM)

- Community detection as a percolation process [Palla05]
- Starting from a *k*-clique, nodes are attached as long as they are reachable through clique adjacency
- *k*-clique community: the union of *k*-cliques that can be reached from one to the other through a sequence of adjacent *k*-cliques

k-clique: complete sub-graph of *k* nodes



CPM applied to static time-dependent snapshots of two social networks [Palla07]

- Los Alamos cond-mat article archive (142 months, 30000 authors)
- Mobile phone-call network (52 weeks, 4 million users)

Community Mapping

G'_t: joint graph comprising the union of links from networks G_t and G_{t+1}

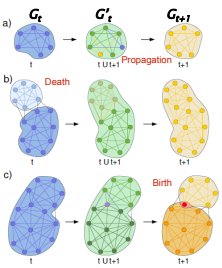
C_t, C'_t: community structure of G_t, G'_t

For each community $C_t^i \in C_t$ and $C_{t+1}^j \in C_{t+1}$ there is one community $C_t'^k \in C'_t$ containing it.

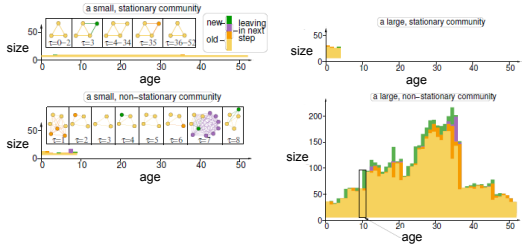
Mapping procedure

1. For each $C_t'^k \in C'_t$ find sets C_t^k and C_{t+1}^k of enclosed communities
2. For each pair $(C_{i,t}^k, C_{j,t+1}^k)$, $C_{i,t}^k \in C_t^k$ and $C_{j,t+1}^k \in C_{t+1}^k$ compute relative overlap
3. Match pairs of communities in descending order of relative overlap

$$rel\_overlap_{i,j}^k = \frac{C_{i,t}^k \cap C_{j,t+1}^k}{C_{i,t}^k \cup C_{j,t+1}^k}$$



Community evolution results in the co-authorship network



Community detection in snapshot graphs and mapping across successive structures

- based on Clique Percolation and community overlap [Palla07]
- **based on Mutual Awareness expansion and interaction correlation [Lin07]**
- based on the membership of core nodes [Wang08]

Community detection based on mutual awareness expansion

Identification and evolution of thematic communities in the Blogosphere [Lin07]

Two actors being aware of each other. E.g. two bloggers leaving comments on each other's blog

Mutual awareness leads to community formation.

- Given a query, construct a time-dependent directed graph where nodes are bloggers and edges their interactions.
- Model mutual awareness expansion with a random walk process and find community structure for each graph *G(V,E)*
  - Compute expected path length (symmetric social distance – ssd) between every pair of nodes
  - Iteratively isolate a set of nodes *S* (community) from *G* so that ssd between members of *S* and $V \setminus S$ is maximized
- Identify community evolution by mapping communities from two subsequent structures

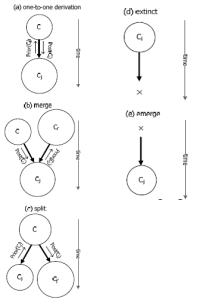
Community mapping using interaction correlation

Each community is represented with a vector in an interaction space, modeling the interactions between all actors

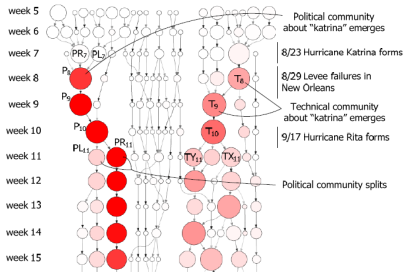
Histogram intersection is used to compute the interaction correlation of two communities

For each community C_i^t find $\text{Post}(C_i^t)$: *the* one that is more similar to it in the future

For each community C_i^{t+1} find $\text{Prior}(C_i^{t+1})$: the one that is more similar to it in the past



Evolution of communities in the "Katrina" –query graph



Community detection in snapshot graphs and mapping across successive structures

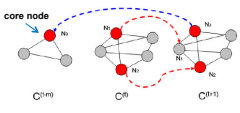
- based on Clique Percolation and community overlap [Palla07]
- based on Mutual Awareness expansion and interaction correlation [Lin07]
- **based on the membership of core nodes [Wang08]**

CoreTracker

Observation: community membership is generally unstable

Focus on representative core nodes to track community evolution [Wang08]

- Community detection in graph snapshots with whichever algorithm
- Identification of core nodes in each community based on *centrality*
- Community C_{t+1} **successor of** C_t if they share a common core vertex **AND** C_{t+1} shares a common core vertex with an ancestor *of* C_t



Phenomena

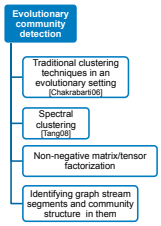
Split: C_t has more than one successor
Mergence: C_t owns more than one predecessor
Birth: C_t has no predecessor
Death: C_t has no successor

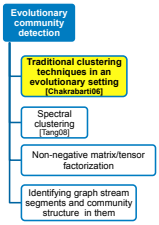
Evolutionary community identification

- Real world data are usually ambiguous and noisy. An algorithm extracting communities at each time-step independently of the other, often results in community structures with high temporal variation.
- Evolutionary community identification methods lead to smoother community evolution, as they utilize the history of the community structure to maximize temporal smoothness

Approaches:

- Traditional clustering techniques in an evolutionary setting
- Spectral clustering
- Non-negative matrix/tensor factorization
- Methods identifying graph stream segments and detecting community structure in them





Traditional clustering techniques in an evolutionary setting

Evolutionary clustering in an online setting [Chakrabarti06]

Simultaneous optimization of two potentially conflicting criteria:
(i) snapshot quality ***sq***, and (ii) history quality ***hq***

At each time-step the framework finds a clustering based on the new similarity matrix M_t and the history so far, which optimizes:

$$sq(C_t, M_t) - cp \cdot hc(C_{t-1}, C_t)$$

Generic framework
Two instantiations are proposed:
•agglomerative hierarchical algorithm
•k-means

Similarity matrix

- Let matrix $F(t)$ store the #features shared between each pair of nodes

The similarity measure should have memory of previous interactions

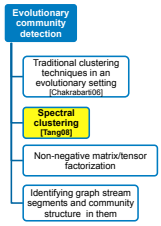
$$F(t) = (1-\beta)\cdot F(t) + \beta \cdot F(t-1), \text{for } t > 0$$

$$S_t(i,j) = \langle r_i, r_j \rangle + |r_i| \cdot |r_j|$$

- *local similarity matrix* : (cosine similarity)

total similarity matrix $M_t(i,j) = a \cdot S_t(i,j) + (1-a) \cdot Sim(i,j,t)$

temporal similarity via time-series correlation



Spectral clustering approach

Spectral clustering uses the spectrum of the graph's similarity matrix to perform dimensionality reduction for clustering in fewer dimensions

Evolutionary spectral clustering applied in multi-mode networks [Tang08]

- m-mode network: $X_{(i)} = \{x_1^i, x_2^i, \ldots, x_{n_i}^i\}, i = 1, \ldots, m$
- Interaction between two modes approximated by interactions between communities: $R_{i,j}^t \approx C^{(i,t)} A_{i,j}^t (C^{(j,t)})^T$, where $C^{(i,t)} \in \{0,1\}^{n_i \times k_i}$: community membership for actors in mode $X_{(i)}$ at t and $A_{i,j}^t$: group interaction matrix

$$\mathbf{F_2}: \min \sum_{t=1}^{\mathbb{T}} \sum_{i<j} w_{ij}^{(i,j)} \| R_{i,j}^t - C^{(i,t)} A_{i,j}^t (C^{(j,t)})^T \|_F^2$$
$$+ \frac{1}{2} \sum_{i=1}^{m} w_b^{(i)} \sum_{t=2}^{\mathbb{T}} \| C^{(i,t)} (C^{(i,t)})^T - C^{(i,t-1)} (C^{(i,t-1)})^T \|_F^2$$
$$s.t. \quad (C^{(i,t)})^T C^{(i,t)} = I_{k_i} \quad \text{spectral relaxation}$$
$$with \quad i = 1, \cdots, m, \quad t = 1, \cdots \mathbb{T}$$

(interaction approximation; temporal smoothness)

The effect of temporal change is adopted as a regularization term

Spectral clustering approach

Spectral clustering uses the spectrum of the graph's similarity matrix to perform dimensionality reduction for clustering in fewer dimensions

Evolutionary spectral clustering applied in multi-mode networks [Tang08]

- m-mode network: $X_{(i)} = \{x_1^i, x_2^i, \ldots, x_{n_i}^i\}, i = 1, \ldots, m$
- Interaction between two modes approximated by interactions between communities: $R_{i,j}^t \approx C^{(i,t)} A_{i,j}^t (C^{(j,t)})^T$, where $C^{(i,t)} \in \{0,1\}^{n_i \times k_i}$: community membership for actors in mode $X_{(i)}$ at t and $A_{i,j}^t$: group interaction matrix

$$\mathbf{F_2}: \min \sum_{t=1}^{\mathbb{T}} \sum_{i<j} w_{ij}^{(i,j)} \| R_{i,j}^t - C^{(i,t)} A_{i,j}^t (C^{(j,t)})^T \|_F^2$$
$$+ \frac{1}{2} \sum_{i=1}^{m} w_b^{(i)} \sum_{t=2}^{\mathbb{T}} \| C^{(i,t)} (C^{(i,t)})^T - C^{(i,t-1)} (C^{(i,t-1)})^T \|_F^2$$
$$s.t. \quad (C^{(i,t)})^T C^{(i,t)} = I_{k_i} \quad \text{spectral relaxation}$$
$$with \quad i = 1, \cdots, m, \quad t = 1, \cdots \mathbb{T}$$

(interaction approximation; temporal smoothness)

- $C^{(i,t)}, t = 1:\mathbb{T}$, are iteratively calculated with an effective algorithm involving eigen-vector calculation until $\mathbf{F_2} < \epsilon$
- Communities are derived from $C^{(i,t)}$ with **K-means**

Non-negative matrix/tensor factorization

- Evolutionary community detection
  - Traditional clustering techniques in an evolutionary setting [Chakrabarti06]
  - Spectral clustering [Tang08]
  - **Non-negative matrix/tensor factorization**
    - **community factorization [Chi07]**
    - FacetNet [Lin08]
    - MetaFac [Lin09]
  - Identifying graph stream segments and community structure in them

Community factorization

Model of community with
- structural aspects: **community graph** representing intra-community interactions
- temporal aspects: **community intensity** representing its activity level over time

Representation of blogosphere at time t as a mixture of community graphs weighted by their intensities – Community Factorization [Chi07]

Let $\mathcal{A} = \{A_1, A_2, \ldots, A_t\} \in \Re^{N \times N \times T}$ be the tensor created by the adjacency matrices of the successive graph snapshots

From each A_t a set of basis dense subgraphs $\boldsymbol{B}_t$ is extracted via graph partitioning

Basis tensor $\mathcal{B}$ is obtained by stacking all $\boldsymbol{B}_t$ together

Community graph: $C_i = \sum_{j=1}^{m} u_{ji} B_j$, where m : # of all basis subgraphs and u_{ji}: weight

Problem formulation: minimization of $\frac{1}{2}\left\| \mathcal{A} - \sum_{i=1}^{k} C_i \right\|^2, C_i = [v_{1i}C_i, \ldots, v_{ti}C_i] \in \Re_+^{N \times N \times T}$

v_{ti}: the intensity of community C_i at time t

Community factorization

Model of community with
- structural aspects: **community graph** representing intra-community interactions
- temporal aspects: **community intensity** representing its activity level over time

Representation of blogosphere at time t as a mixture of community graphs weighted by their intensities – Community Factorization [Chi07]

Let $\mathcal{A} = \{A_1, A_2, \ldots, A_t\} \in \Re^{N \times N \times T}$ be the tensor created by the adjacency matrices of the successive graph snapshots

From each A_t a set of basis dense subgraphs $\boldsymbol{B}_t$ is extracted via graph partitioning

Basis tensor $\mathcal{B}$ is obtained by stacking all $\boldsymbol{B}_t$ together

Community graph: $C_i = \sum_{j=1}^{m} u_{ji} B_j$, where m : # of all basis subgraphs and u_{ji}: weight

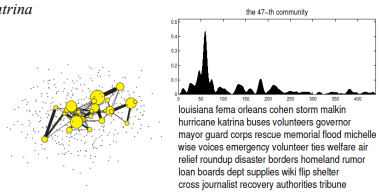
equivalent to $\frac{1}{2}\left\| \mathcal{A} - (\mathcal{B} \times_3 U) \times_3 V^T \right\|^2, U \in \Re_+^{m \times k}\ V \in \Re_+^{T \times k}$

Community factorization

U, V are identified by non-negative matrix factorization

Regularization terms are also inserted to smooth community temporal trends (evolution of intensity over time)

Community graph and temporal trends for community related to hurricane Katrina



Identification of graph stream segments and community structure

- Evolutionary community detection
  - Traditional clustering techniques in an evolutionary setting [Chakrabarti06]
  - Spectral clustering [Tang08]
  - Non-negative matrix/tensor factorization
  - **Identifying graph stream segments and community structure in them**
    - GraphScope [Sun07]
    - Stream-Group [Duan09]

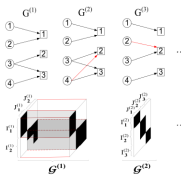
Problem definition: given a graph stream, find good change points in time to *segment* it, and identify communities within each segment.

- Evolutionary community detection
  - Traditional clustering techniques in an evolutionary setting [Chakrabarti06]
  - Spectral clustering [Tang08]
  - Non-negative matrix/tensor factorization
  - **Identifying graph stream segments and community structure in them**
    - **GraphScope [Sun07]**
    - Stream-Group [Duan09]

GraphScope

Method applied on unweighted undirected bipartite graphs [Sun07]

Operates without parameters based on Minimum Description Length (MDL)



Considering the graph as a binary adjacency matrix with "1" denoting the presence of a link between two nodes, the goal is to organize the graph into homogeneous sub-matrices with low entropy and compress them separately.

Aim: the minimization of total encoding cost

$$\text{cost} = \sum_{segment_t} C^{(segment_t)} = \sum_{s_t} C^{(s_t)} = \sum_{s_t} \log(t_{s_t+1} - t_{s_t}) + C_p^{(s_t)} + C_g^{(s_t)}$$

Graph encoding cost

$$C_g^{(s_t)} = \sum_{p=1}^{k_{s_t}} \sum_{q=1}^{l_{s_t}} (\log|E|_{p,q}^{(s_t)} + |G_{p,q}^{(s_t)}| \cdot H(G_{p,q}^{(s_t)}))$$

Partition encoding cost

$$C_p^{(s_t)} = \log m + \log n + \log k_{s_t} + \log l_{s_t} + mH(P) + nH(Q)$$

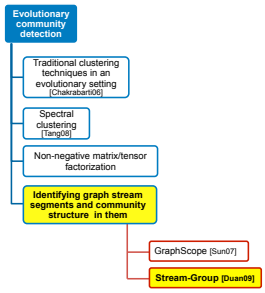
GraphScope

- Depending on the lowest encoding cost, each new graph is included in the current segment or initiates a new segment
- For each segment, source and destination nodes are partitioned separately permuting the matrix's rows and columns so that the encoding cost is decreased

Finally

Compression with **GraphScope** achieves in fitting the graph in less than 4% of the original space

Data are organized into few homogenous **communities**



Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that random walker will stay at *i* when starting from *j*
  - relevance scores between within community nodes are usually higher than the ones between different communities
- an extension of *modularity* to evaluate the goodness of a partition

Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that ra… when starting from *j*
  - relevance scores between within c… ly higher than the ones between different communiti…
- an extension of *modularity* to evaluate the goodness of a partition

#edges falling within communities minus the expected number in an equivalent random network

Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that random walker will stay at *i* when starting from *j*
  - relevance scores between within community nodes are usually higher than the ones between different communities
- an extension of *modularity* to evaluate the goodness of a partition

Procedure

1. For each new graph arrival, identify its community structure
2. Compute similarity between new structure and current segment's structure
3. If similarity over threshold, the community structure of the current segment is updated incrementally with the new data
4. else, a new segment is initiated

Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that random walker will stay at *i* when starting from *j*
  - relevance scores between within community nodes are usually higher than the ones between different communities
- an extension of *modularity* to evaluate the goodness of a partition

Procedure

1. For each new graph arrival, identify its community structure
2. Compute similarity between new structure and current segment's structure
3. If similarity over threshold, the community structure of the current segment is updated incrementally with the new data
4. else, a new segment is initiated

1. Find compact groups of nodes via an heuristic step
2. Merge groups while goodness indicator increases

Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that random walker will stay at *i* when starting from *j*
  - relevance scores between within community nodes are usually higher than the ones between different communities
- an extension of *modularity* to evaluate the goodness of a partition

$$Sim(I, I') = \frac{P_r(I') + P_r(I)}{2|V|}, \qquad P_r(I') = \sum_p \max_{c \in I'} |I_p \cap c|$$

Procedure

1. For each new graph arrival, identify its community structure
2. Compute similarity between new structure and current segment's structure
3. If similarity over threshold, the community structure of the current segment is updated incrementally with the new data
4. else, a new segment is initiated

Stream–Group

Stream–Group: applied on dynamic weighted directed graphs [Duan09]

Uses:

- *Random Walk with Restart* to compute the graph's relevance matrix *R*
  - r_{ij} expresses the probability that random walker will stay at *i* when starting from *j*
  - relevance scores between within community nodes are usually higher than the ones between different communities
- an extension of *modularity* to evaluate the goodness of a partition

Procedure

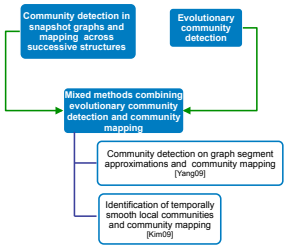
1. For each new graph arrival, identify its community structure
2. Compute similarity between new structure and current segment's structure
3. If similarity over threshold, the community structure of the current segment is updated incrementally with the new data
4. else, a new segment is initiated

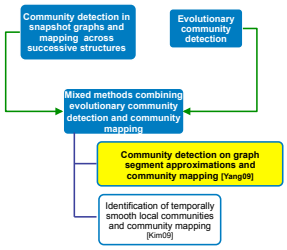
Merging step of the community detection alg using as seed the **Intersection** of the partitions of the current segment and the new graph

Mixed methods combining evolutionary community detection and community mapping

Approaches:

- Identification of temporally smooth local communities and then mapping across successive community structures [Kim09]
- Community detection on graph segment approximations and then mapping across successive community structures [Yang09]

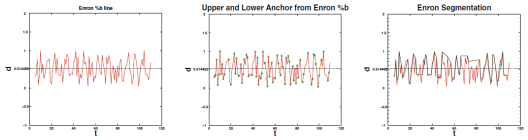




Tracing the Timeline of Networks

Changes in degree, neighbors

- Identifies changepoints in the graph stream by measuring the distance between nodes [new–born, deceased, stable] of subsequent snapshots [Yang09]
- Generates smooth approximations of graph segments to address the problem of **noise**, by iteratively adding to the approximation graph edges whose nodes have small distance

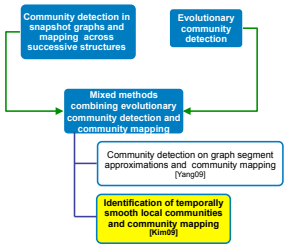


Segmentation and smoothing for the Enron dataset

Tracing the Timeline of Networks

Changes in degree, neighbors

- Identifies changepoints in the graph stream by measuring the distance between nodes [new-born, deceased, stable] of subsequent snapshots [Yang09]
- Generates smooth approximations of graph segments to address the problem of **noise**, by iteratively adding to the approximation graph edges whose nodes have small distance
- *Community detection method requiring definition of **three** parameters*
  - ✓identification of all *k-cliques* in each approximate graph
  - ✓multiple community memberships are solved assigning the respective nodes to the community with which they have most interactions (*weight*)
  - ✓each community attract its neighboring nodes whose *weight* to the community is over a *judgment threshold*
  - ✓communities that are tightly connected (with respect to a *weight threshold*) are merged
- *Community correlation and evaluation*
  - ✓this method does not apply community mapping directly, but rather evaluates the smoothness of successive community structures **by calculating their correlation considering both edge and node overlap**



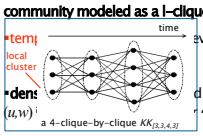
A Particle – and Density-based Method

Removes the constraint for identical number of communities between successive structures [Kim09]

Creates a *t-partite* graph linking nodes belonging to subsequent graph snapshots based on a similarity function

E.g. considered similar when sharing a neighbor

community modeled as a l-clique-by-clique (l-KK)



•temp … evel with cost embedding

$$COS_N = a \cdot (d_o(u,w) - d_t(u,w)) + (1-a) \cdot (d_{t-1}(u,w) - d_t(u,w))$$

•dens… d on σ'_t, an extension of structural similarity σ_t [$\sigma_t(u,w)$ … f common neighbors between u,w]

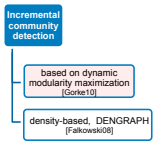
▪automatically determines parameter ε with *modularity optimization*

Community mapping based on the density of links between local clusters with an heuristic alg maximizing mutual information

$$\sigma'_t(u,w) = a \cdot (\sigma_t(u,w) - \sigma_{t-1}(u,w)) + \sigma_{t-1}(u,w)$$

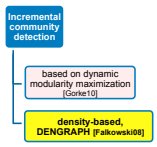
Incremental community detection

The community structure of successive time-steps is incrementally adjusted based on graph modification operations.



- ▪Graph modification operations (e.g. new vertex or edge) arrive as a stream
- ▪Community structure is **updated** rather than recalculated (**improves efficiency**)

Naïve approach compared to evolutionary community identification



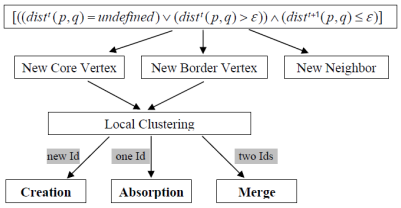
DenGraph

*Incremental **density-based** clustering method* [Falkowski08]

- ▪Let $min\{num\_interactions_{i\to j}, num\_interactions_{i\to j}\} = k_{i,j}$
- ▪Node-distance matrix, where: $dist(i,i) = 1$ and $dist(i,j) = 0$ if $k_{i,j} = 0$, or else $dist(i,j) = 1 / k_{i,j}$
- ▪Clusters are built by connecting adjacent (μ, ε)-neighborhoods [Ester96]
- ▪A (μ,ε)-neighborhood is built around a **core node** and includes at least μ nodes within a radius of ε
- ▪**Border nodes** are included in a neighborhood but are not cores
- ▪**Noise nodes** are not included in any neighborhood

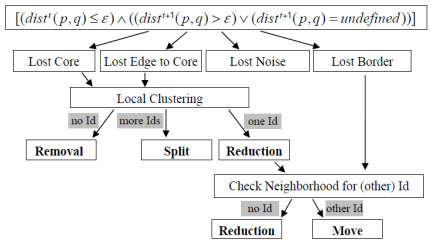
DenGraph

Effects of increased proximity among nodes upon the community



DenGraph

Effects of decreased proximity or node/edge removal



Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

Block 4: Mining evolving social data (Athena Vakali)

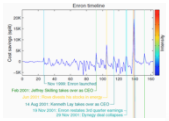
- ✓ Structure and Models
- ✓ Community Detection in Evolving Social Graphs
- ▪ Applications of Mining Evolving Social Data

Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Applications of Mining Evolving Social Data

The results of community detection, or different mining techniques, on evolving social data can be exploited in applications:



event detection
diagram from [Sun07]

clustering of users
exploiting the time dimension

social network analysis
image from [Touchgraph]

trend detection
image from [Trendsmap]

[Touchgraph] http://www.touchgraph.com/TGFacebookBrowser.html
[Trendsmap] http://trendsmap.com/

Social network analysis

- Data mining can offer new insights in the analysis of the structure and evolution of social networks.
  - *Important technique* : **evolving community detection**, as communities constitute meaningful units of organization
- The methods we mentioned so far have presented research results for data networks derived from the following sources…

Social network analysis

Source	Community detection method
synthetic datasets	Chi07, Duan09, Gorke10, Kim09, Lin08, Tang08, Yang09
benchmark datasets (e.g. dolphin's network, IEEE VAST dataset)	Chi07, Yang09
article online archives (e.g. DBLP, arXiv)	Gorke10, Kim09, Lin08, Palla07, Tang08, Wang08, Yang09
mobile phone call networks	Palla07, Sun07, Wang08, Yang09
imdb actor collaboration dataset	Wang08
Enron e-mail dataset	Duan09, Falkowski08, Sun07, Tang08, Wang08,Yang09
blog data	Chi07, Lin07, Lin08
Flickr	Chakrabarti06
mobile device proximity networks	Sun07
company call networks	Yang09
football match schedules	Kim09
department e-mail dataset	Gorke10
Digg	Lin09

Event detection

Definition of event

closely related with time

- *the* ***information flow*** *between a group of* ***social actors*** *on a specific* ***topic*** *over a certain* ***time period*** [Zhao07]
- *occasions that take place at a* ***specific time*** *and* ***location****, for instance concerts, parties, etc.* [Quack08]

Graph segmentation community detection methods [GraphScope, Stream-Group] can identify significant change-timepoints in the stream which constitute *events*.

- Event detection from social data streams (*triples of actor, text, time*) exploring features in three dimensions: textual content, social, and temporal [Zhao07]

1. text representation with vectors using TF-TD and application of the graph-cut clustering algorithm [Shi00], resulting in topic clusters
2. for each topic: timeline segmentation into meaningful intervals based on fluctuations in the *intensity* of topic-related discussion
3. for each topic: calculation of (i) *information flow* pattern between pairs of actors as vectors and (ii) *dynamic time warping-based similarity* between pairs of information flow patterns
4. creation of a network with pairs of actor as nodes and calculated similarities as edges, and application of graph-cut clustering [Shi00]

Event detection

Definition of event

closely related with time

- *the* ***information flow*** *between a group of* ***social actors*** *on a specific* ***topic*** *over a certain* ***time period*** [Zhao07]
- *occasions that take place at a* ***specific time*** *and* ***location****, for instance concerts, parties, etc.* [Quack08]

Graph segmentation community detection methods [GraphScope, Stream-Group] can identify significant change-timepoints in the stream which constitute *events*.

Dataset : geo-tagged photos from Flickr grouped in areas [Quack08]

1.Image clustering: computation of a *dissimilarity matrix* combining visual and textual features (tags), and application of a hierarchical agglomerative clustering algorithm

2.Classification of clusters in **objects** or **events** using an ID3 decision tree and two extra features for each cluster:

- the number of days covered by its photos (defined by the photos' timestamps)
- the number of users who took the photos

Main idea: objects, such as landmarks, are photographed by many users throughout the year, whereas **events** usually last one or two days and are covered by fewer users

Trend detection

- Social data fluctuate in their structure and frequency as they evolve. At each time-period there are some topics, images, tags, etc, that are most popular amongst users (**trends**). **Data mining can be used for detecting trends in evolving social data.**
- Trends can be identified *globally* or even *locally* (within communities) and they usually indicate what interests users the most at a given time

Trend identification in Twitter

- **Twitter** : popular microblogging website where users are allowed to post short messages (up to 140 chars) and "*follow*" the posts of others
- Rich source of rapidly evolving social data which are also **public**. Suitable for *trend detection*
- Recently, there have been many attempts to statistically analyze Twitter data
- Evolving Twitter data to identify trending keywords for different weekdays [Java07]
- Microblogging as a form of electronic word-of-mouth for sharing consumer opinions concerning brands. Sentiment identification performed in Twitter posts to identify trending sentiments about brands [Jansen09].

Trend detection

- Social data fluctuate in their structure and frequency as they evolve. At each time-period there are some topics, images, tags, etc, that are most popular amongst users (**trends**). **Data mining can be used for detecting trends in evolving social data.**
- Trends can be identified *globally* or even *locally* (within communities) and they usually indicate what interests users the most at a given time

Trend identification in the blogosphere

Traditional approach: calculation of the frequency of appearance of each term in all blogs for each time-step

However, in [Chi06] different emphasis is put on every blog depending on its overall amount of contribution to the trend

Data represented as a combination of information capturing: (i) temporal changes in them and (ii) characteristics of individual bloggers

- **Method based on SVD:** produces *scalar eigen-trends* capturing overall trends, and *authority scores* representing the contribution of each blog to trends
- **Method based on HOSVD:** applied on keyword-specific blog-citation graphs to produce *structural eigen-trends*, *hub scores* and *authority scores* capturing structural changes

Resemblance to HITS

Clustering of users exploiting the dimension of time

Social data can be analyzed for automatic synthesis of *user profiles*

Case study: **Social Tagging Systems (STS)**

Clustering of users in **STS** according to **topics of interest** and the **time locality of tagging activity** [Koutsonikola09]

e.g. *a user who tagged a set of photos depicting sports is probably interested in sports. However, if his tagging activity took place during the Olympic games, maybe he is simply interested in the Olympics and is not a regular sports fan.*

- ***Segmentation of time in frames***
- A user is related to a given tag if he has assigned at least one semantically close tag. The topic-distance between two users is calculated based on the similarity of their relations to all involved tags.
- Time-similarity between a user and a tag is calculated with the *cosine coefficient*. The time-distance between two users is calculated considering their similarity over all timeframes.
- Topic-based clustering with K-means, then refinement with time criterion

References (Block 4) (1 / 3)

[Au Yeung09] Au Yeung, C.M., Gibbins, N., and Shadbolt, N. 2009. Contextualising Tags in Collaborative Tagging Systems. In Proceedings of 20th ACM Conference on Hypertext and Hypermedia, pp. 251–260.

[Chakrabarti06] Chakrabarti, D., Kumar, R., and Tomkins, A. 2006. Evolutionary clustering. In Proceedings of the 12th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining . KDD '06. ACM, New York, NY, 554–560.

[Chi06] Chi, Y., Tseng, B. L., and Tatemura, J. 2006. Eigen-trend: trend analysis in the blogosphere based on singular value decompositions. In Proceedings of the 15th ACM international Conference on information and Knowledge Management . CIKM '06. ACM, New York, 68–77.

[Chi07] Chi, Y., Zhu, S., Song, X., Tatemura, J., and Tseng, B. L. 2007. Structural and temporal analysis of the blogosphere through community factorization. In Proceedings of the 13th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining. KDD '07. ACM, New York, NY, 163–172.

[Duan09] Duan, D., Li, Y., Jin, Y., and Lu, Z. 2009. Community mining on weighted directed graphs. In Proceeding of the 1st ACM international Workshop on Complex Networks Meet information & Knowledge Management . CNIKM '09. ACM, New York, NY, 11–18.

[Ester96] Ester, M., Kriegel, H.-P., Sander, J., and Xiu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd international Conference on Knowledge Discovery and Data Mining. KDD '96. AAAI Press, 226–231.

[Falkowski06] Falkowski, T. ,Bartelheimer, J., and Spiliopoulou., M. 2006. Community Dynamics Mining. In Proceedings of the 14th European Conference on Information Systems. ECIS '06.

[Falkowski08] Falkowski, T., Barth, A., and Spiliopoulou M. 2008. Studying Community Dynamics with an Incremental Graph Mining Algorithm. In Proceedings of the 14th Americas Conference on Information Systems. AMCIS.

[Fortunato07] Fortunato, S. & Castellano, C. 2007. Community structure in graphs. arXiv:0712.2716v1.

[Girvan 02] Girvan, M. & Newman, M. E. J. 2002. Community structure in social and biological networks. In Proceedings of the National Academy of Sciences of the United States of America, 99(12):7821–7826.

References (Block 4) (2/3)

[Gorke10] Görke, R., Maillard, P., Staudt, C., and Wagner, D. 2010. Modularity-Driven Clustering of Dynamic Graphs. In Proceedings of the 9th International Symposium on Experimental Algorithms. SEA' 10), Springer, 436–448.

[Hotho06] Hotho, A., Robert, J., Christoph, S., and Gerd, S. 2006. Emergent Semantics in BibSonomy. GI Jahrestagung Vol. P-94, 305–312. Gesellschaft fr Informatik.

[Jansen09] Jansen, B. J., Zhang, M., Sobel, K., and Chowdury, A. 2009. Twitter power: Tweets as electronic word of mouth. J. Am. Soc. Inf. Sci. Technol. 60, 11 (Nov. 2009), 2169–2188.

[Java07] Java, A., Song, X., Finin, T., and Tseng, B. 2007. Why we twitter: understanding microblogging usage and communities. In Proc. of the 9th WebKDD and1 st SNA-KDD Workshop on Web Mining and Social Network Analysis . WebKDD/SNA-KDD '07. ACM,, 56–65.

[Kim09] Kim, M. and Han, J. 2009. A particle-and-density based evolutionary clustering method for dynamic networks. Proc. VLDB Endow. 2, 1, 622–633.

[Koutsonikola09] Koutsonikola, V., Vakali, A., Giannakidou, E., and Kompatsiaris, I. 2009. Clustering of Social Tagging System Users: A Topic and Time Based Approach. In Proceedings of the 10th international Conference on Web information Systems Engineering. G. Vossen, D. D. Long, and J. X. Yu, Eds. Lecture Notes In Computer Science, vol. 5802. Springer-Verlag, Berlin, Heidelberg, 75–86.

[Lin08] Lin, Y., Chi, Y., Zhu, S., Sundaram, H., and Tseng, B. L. 2008. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In Proceedings of the 17th international Conference on World Wide Web. WWW '08. ACM, 685–694.

[Lin07] Lin, Y., Sundaram, H., Chi, Y., Tatemura, J., and Tseng, B. L. 2007. Blog Community Discovery and Evolution Based on Mutual Awareness Expansion. In Proceedings of the IEEE/WIC/ACM international Conference on Web intelligence. Web Intelligence. IEEE Computer Society, Washington, DC, 48–56.

[Mika05] Mika, P. 2005. Ontologies Are Us: A Unified Model of Social Networks and Semantics. In Proceedings of the 4th international SemanticWeb Conference. ISWC'05. Springer Berlin /Heidelberg, pp. 522–536.

References (Block 4) (3/3)

[Palla05] Palla, G., Derényi, I., Farkas, I., and Vicsek, T. 2005. Uncovering the overlapping community structure of complex networks in nature and society. Nature 435, 814–818.

[Palla07] Palla, G., Barabási, A.-L., and Vicsek, T. 2007. Quantifying social group evolution. Nature 446, 664–667.

[Quack08] Quack, T., Leibe, B., and Van Gool, L. 2008. World-scale mining of objects and events from community photo collections. In Proceedings of the 2008 international Conference on Content-Based Image and Video Retrieval . CIVR '08. ACM, New York, NY, 47–56.

[Shi00] Shi, J. and Malik, J. 2000. Normalized cuts and image segmentation. IEEE TPAMI 22(8):888–905.

[Sun06] Sun, J., Tao, D., and Faloutsos, C. 2006. Beyond streams and graphs: dynamic tensor analysis. In Proceedings of the 12th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining . KDD '06. ACM, New York, NY, 374–383.

[Sun07] Sun, J., Faloutsos, C., Papadimitriou, S., and Yu, P. S. 2007. GraphScope: parameter-free mining of large time-evolving graphs. In Proceedings of the 13th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining . KDD '07. ACM,, 687–696.

[Tang08a] Tang, L., Liu, H., Zhang, J., and Nazeri, Z. 2008. Community evolution in dynamic multi-mode networks. In Proceeding of the 14th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining . KDD '08. ACM, New York, NY, 677–685.

[Tang08b] Tong, H., Papadimitriou, S., Yu, P. S., and Faloutsos, C. 2008. Proximity tracking on time-evolving bipartite graphs. In Proceedings of the 9th SIAM international Conference on Data Mining . SDM '08. 704–715.

[Yang09] Yang, S., Wu B., Wang, B. 2009. Tracking the Evolution in Social Network: Methods and Results. Complex (1): 693–706.

[Wang08] Wang, Y., Wu, B., and Du, N. 2008. Community Evolution of Social Network-Feature, Algorithm and Model. arXiv: 0804.4356.

[Zhao07] Zhao, Q., Mitra, P., and Chen, B. 2007. Temporal and information flow based event detection from social text streams. In Proc. of the 22nd National Conference on Artificial intelligence – Volume 2. Aaai Conference On Artificial Intelligence. AAAI Press, 1501–1506.

End of Block 4

Thank you!

Questions?

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction
- Approaches

(Ernestina Menasalvas)

Block 6: Conclusions and Outlook

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction
- Approaches

(Ernestina Menasalvas)

Block 6: Conclusions and Outlook

Introduction and Motivation

Challenge:

•a stream is theoretically infinite so cannot be materialized.

Data have to be processed in a single pass using little memory.

Based on this restriction one can identify two divergent objectives:

1.the analysis should produce comprehensive and exact results and detect changes in the data as soon as possible.

2.The single pass demand together with **resource limitations** allow only to perform the analysis on an approximation of the stream or a window (finite subset of the stream).

Introduction and Motivation

Data streams are mostly generated or sent to resource constrained computing environments:

- Data generated on-board astronomical spacecrafts
- Data generated in sensor networks. The additional constraint that sensor nodes consume their energy rapidly with data transmission
- Data received in resource-constrained environment represents a different category of applications:
  - Personal Digital Assistants PDAs: users might request sheer amounts of data of interest to be streamed to their mobile devices. Storing and retrieving these huge amounts of data are also infeasible in such an environment

Applications

•monitoring physiological data streams obtained from wearable sensing devices. Such monitoring can be either for:

- applications for pervasive healthcare management,
- Applications for seniors,
- emergency response personnel,
- soldiers in the battlefield
- or athletes

•Onboard analysis of data streams: data generation would exceed the bandwidth to transfer these streams of data to ground stations for analysis. They necessitate the need for onboard analysis of data streams.

Applications

•Wearable sensors available in theMarket

- SenseWear Armband from BodyMedia
- Wearable West
- LifeShirt Garment from Vivometrics

•SenseWear armband can measure heat flux, accelerometer, galvanicskin response, skin temperature,near body temperature

•Arm band can store up to about 5 days of data.

•Detecting emerging patterns in soldiers, elderly individuals, animals…

Applications

•MineFleet: A Vehicle Data Stream Management and Mining Software System

- On-board Module:
  - Continuous data streams from the vehicle data bus
- Onboard data stream mining
  - Communicates with a remote control station. Privacy management
- Central control station:
  - Data Management, Data mining, Communicates with the on-board modules over wireless networks, Privacy management

Applications

•Vehicle Data Stream Mining [Kargupta]

- **Vehicle Health Monitoring and Maintenance:**
  - Detecting unusual behavior for a subsystem
- **Fuel Consumption Analysis:**
  - Is the vehicle burning fuel efficiently? Identify influencing factors
  - Detect influence of driver behavior on gas mileage
- **Driver Behavior Monitoring:**
  - Route monitoring: Fixed and variable routes
  - Direct Cost Issues: e.g. Idling, braking habits, Safety Issues
- **Vehicle location related services**
- **Vehicular network security and privacy management**

Computational resources affected

Stream mining affects settings in:

- memory,
- processing cycles,
- Communication bandwidth and
- battery.

•Stream mining algorithms:

- typically linear or sub-linear algorithms
- characterized by being space efficient.

•BUT:

- Most of these algorithms are not designed with regard to adaptation to resource availability

Issues (Gaber)

- the input,
- output, and
- processing settings of an algorithm
  - could be changed according to measurements of resource availability in a time frame:
- Find:
  - resource consumption patterns and
  - algorithm settings.

•Challenge:

- Limited computational resources
- Limited bandwidth
- Change of the user's context

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction
- Approaches

(Ernestina Menasalvas)

Block 6: Conclusions and Outlook

Approaches

1. Based on user specified quality requirements the algorithm derives resource requirements, i.e., the memory needed for managing stream approximations in order to guarantee the requested quality.
2. Input adaptation:
   - load shedding and data synopsis creation using wavelets
- Some drawbacks:
  - solution is through a specific algorithm,
  - does not handle the situations where more than one resource is constrained, may not apply to all kinds of resources,
  - usually requires some extra processing such as sampling

Memory availability (Frakle)

2 ways of putting resource and quality awareness:

1.Claim for specific quality requirements and deduce the needed resources to achieve this quality.

2.Limit the resources provided for processing and deduce the achievable quality.

- Q –> R ?
- R –> Q ?

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction
- Approaches: Association rules

(Ernestina Menasalvas)

Block 6: Conclusions and Outlook

Association mining. [Nan Jiang and Le Gruenwald]

Issues to take into account:

1.Data Processing method

2.Memory management method

3.Data structure to keep itemsets

4.One pass algorithm

5.Maintenance and generation

Data Processing model

Landmark model

- mines all frequent itemsets over the entire history of stream data from a specific time point called landmark to the present.
- not suitable for applications where people are interested only in the most recent information of the data streams, such as in the stock monitoring systems

Damped model, (Time–Fading model), mines frequent itemsets in stream data in which each transaction has a weight and this weight decreases with age.

- Older transactions contribute less weight toward itemset frequencies. Different weights for new and old transactions.
- suitable for applications in which old data has an effect on the mining results, but the effect decreases as time goes on.

Data Processing model

Sliding Windows model finds and maintains frequent itemsets in sliding windows. Only part of the data streams within the sliding window are stored and processed at the time when the data flows in.

- The size of the sliding window may be decided according to applications and system resources.
- The mining result of the sliding window method totally depends on recently generated transactions in the range of the window;
- all the transactions in the window need to be maintained in order to remove their effects on the current mining results when they are out of range of the sliding window.

Memory management

- Classical association rule mining algorithms on static data collect the count information for all itemsets and discard the non–frequent itemsets and their count information after multiple scans of the database. when we mine association rules in stream data:
  1. there is not enough memory space to store all the itemsets and their counts when a huge amount of data comes continuously.
  2. the counts of the itemsets are changing with time when new stream data arrives. Therefore, we need to collect and store the least information possible
- Some methods uses sizes of itemsets such as 3 or 2 to generate only this itemsets.

Memory management [Nan Jiang and Le Gruenwald]

An efficient and compact data structure is needed to store, update and retrieve the collected information:

- bounded memory size and
- huge amounts of data streams coming continuously.

Failure in developing such a data structure will largely decrease the efficiency of the mining algorithm

The data structure needs to be incrementally maintained.

- it is not possible to rescan the entire input due to the huge amount of data and requirement of rapid online querying speed.

One Pass Algorithm to Generate Association Rules [Nan Jiang and Le Gruenwald]

Association rules can be found in two steps:

1.finding large itemsets (support is greater than user specified support) for a given threshold support and

2. generate desired association rules for a given confidence

Frequent itemsets [Nan Jiang and Le Gruenwald]

Should we use an exact or approximate algorithm to perform association rule mining in data streams?

Can its error be guaranteed if it is an approximate algorithm?

How to reduce and guarantee the error?

What is the tradeoff between accuracy and processing speed?

Is data processed within one pass?

Can this algorithm handle a large amount of data?

Up to how many frequent itemsets can this algorithm mine?

Can this algorithm handle concept drifting and how?

Association generation and maintenance [Nan Jiang and Le Gruenwald]

Mining association rules involves a lot of memory and CPU costs.

This is especially a problem in data streams since the processing time is limited to one online scan.

Approaches of the static world: frequent updating

data stream environment, stream data are added continuously, and therefore, if we update association rules too frequently, the cost of computation will increase drastically.

Some methods assume little concept drifting, that is to say the change of data distribution is relatively small.

Resources awareness

Some approaches:

- Gaber: AOG, which uses a control parameter to control its output rate according to memory, time constrains and data stream rate (see later further)
- Teng et al.: RAMDS algorithm to not only reduce the memory required for data storage but also retain good approximation of temporal patterns given limited resources like memory space and computation power

CFI-Stream: –Nan Jiang, Le Gruenwald

Mining closed frequent itemsets over datastreams:

- computes and maintains closed itemsets online and incrementally,
- can output the current closed frequent itemsets in real time based on users' specified thresholds.
- time and space efficient, has good scalability as the number of transactions processed increases and adapts very rapidly to the change in datastreams.

Addition procedure of the algorithm [Nan Jiang and Le Gruenwald]

Checks if X is in the current closed itemsets set C.

- If X is in C, it updates X's support, and for all X's subsets Y belonging to C, it updates Y's supports.
- Else, if X is not in C and X has been included by at least one transaction in the original transaction set, it checks whether it is a closed itemset for itself and all its subsets; and it updates the associated supports for all the closed itemsets.
  - If X is a newly arrived closed itemset and does not exist in the DIU tree, the algorithm adds it as a new node to the DIU tree.
  - else, if X is the added transaction itself, it adds X into the closed itemset (lines 10–15); if X is the subset of added transaction, a closure checking is performed

Deletion phase of the algorithm [Nan Jiang and Le Gruenwald]

When an itemset X leaves the current sliding window CFI-Stream:

- checks if X is in the current closed itemsets set C and its count is greater or equal to two; if so, it updates X's support and X's subsets' support belonging to C
- Otherwise, it checks the itemset X and all its subsets which are in the current closed itemset C to see whether they are still closed itemsets and updates the support for all its subsets that are in the current closed itemsets
  - If the subset Y exists in transaction, Y should keep
  - Otherwise a closure check for the subset Y is performed

Performance analysis [Nan Jiang and Le Gruenwald]

Time and space efficiency independent of support information, and

it can adapt to the concept-drifting in data streams.

better performance than other state-of-the-art approaches in

terms of both time and space overhead

- especially when the minimum support is low,
- and the dataset is dense

COFI-tree mining M. [El-Hajj and O.R. Zaiane]

FP-trees used in the mining process can all fit in memory.

COFI algorithm—as an alternative to the FP-growth algorithm

COFI consists of three main phrases:

- the construction of an FP-tree representing the original database,
- The construction of a COFI-tree (Co-Occurrence Frequent Item tree) for each frequent item, and
- the mining of frequent patterns from each COFI-tree.

In the first phrase, a global FP-tree is constructed in the same way as in the FP-growth algorithm. Thus 2 database scan are required—one scan for finding the frequency of each item and another scan for building the FP-tree

COFI tree

the COFI-tree contains:

1. the item,
2. its frequency count and
3. its participation counter. This counter is initialize to 0, and is incremented every time the node is raversed/ participated. At the end of the mining process for the COFI-tree of x, the value of this counter is equal to its frequency count. Note that, the COFI algorithm requires at most two trees (i.e., the global FP-tree and the COFI-tree for a specific item) to co-exist at any time during the mining process, whereas FP-growth usually keeps more than two trees.

Carson Kai-Sang Leung, Dale A. Brajczuk, Jialiang Yu

to reduce memory consumption required by FP-growth, authors replace the first step of FP-streaming by applying the COFI algorithm (instead of the FP-growth algorithm) to find "frequent"patterns.

Such a replacement of the algorithm in the first step of FP-streaming leads to both an increase and a decrease in memory consumption in different stages of the mining process.

On the positive side, the use of COFI algorithm reduces memory consumption by avoiding recursive projections and constructions of FP-trees. Thus, instead of multiple FP-trees, at most two trees (namely, the global FP-tree and a COFI-tree for an item x) need to co-exist during the mining process.

However, on the negative side, the construction of a COFI-tree for item x requires more memory space than the construction of an FP-tree for the same item x

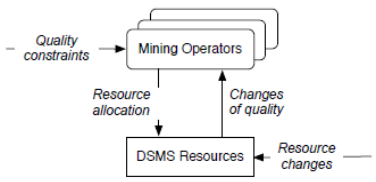
Frequent itemsets FP tree (Franke)

Han et al., 2000: FP-tree

Giannella et al., 2003:

- extension to mine streaming data in a time-sensitive way.
- tilted time window tables represents window-based counts of the itemsets. Allow to maintain summaries of frequency information of recent transactions in the data at a finer granularity
- Update the extended FP tree in batches: accumulate incoming transactions until enough transactions of the stream have arrived. Then, the transactions of the batch are inserted into the tree.
- Mining the tree: modification of the FP-growth algorithm that uses the tilted window table. The original approach assumes that there is enough memory available to deliver results in any required quality

Franke et al approach



1. Tries to calculate memory needed for the quaility required
2. Adapts operators to the memory available

Franke. Adapting dynamically the tree size

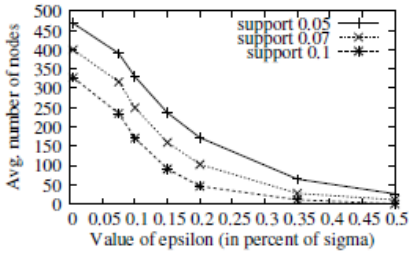
Adapting ϵ

- $f < 0.85$: Decrease ϵ by ten percent. Use this ϵ when processing the following batches.
- $0.85 < f < 1.0$: The value of ϵ remains fixed.
- $f > 1.0$: Increase ϵ by ten percent. Conduct tail pruning at the TTWTs of each node in the pattern tree anddrop all nodes with empty TTWTs. Repeat these steps as long as $f > 1.0$.

Adapting σ

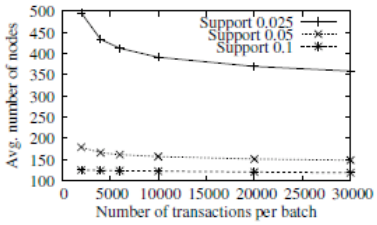
- As σ does not influence the size of the tree directly, ϵ / σ remains the same. That is why the user does not provide a fixed value of σ, but rather claims for a certain ϵ / σ that should be guaranteed. Again, the user may also specify a lower bound for the value of σ.

Franke. Results



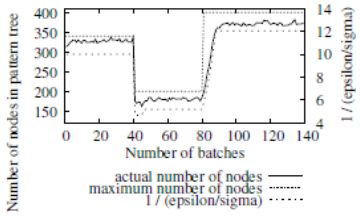
(a) Varying support and epsilon

Franke. Results



(b) Varying support and batch size

Franke. Results



(c) Resource awareness

Memory constraints. Sequential patterns

Teng et al., 2004 :RAM-DS

- Resource-Aware Mining for Data Streams
- uses a wavelet-based approach to control the resource requirements.
- mainly concerned with mining temporal patterns
- the method can only be used in combination with a certain regression-based stream mining algorithm proposed by the same authors.
- Although the proposed algorithm for mining temporal patterns is resource-aware, it is not resource-adaptive and does not provide guarantees for the quality of the mining results

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction
- Approaches: Classification

(Ernestina Menasalvas)

Block 6: Conclusions and Outlook

H. Kargupta. Decision trees

Orthogonal decision trees (ODTs) offer an effective way to construct a redundancy-free, accurate, and meaningful representation of large decision-tree-ensembles:

1. first construct an algebraic representation of trees using multivariate discrete Fourier bases.
2. The new representation is then used for eigen-analysis of the covariance matrix generated by the decision trees in Fourier representation. Converts the corresponding principal components to decision trees.
3. These trees are functionally orthogonal to each other and they span the underlying function space. These orthogonal trees are in turn used for accurate (in many cases with improved accuracy) and redundancy-free (in the sense of orthogonal basis set) compact representation of large ensembles.

ODT. Kargupta: Application weareable

Variables sensed:

- Heat flux: The amount of heat dissipated by the body.
- Accelerometer: Motion of the body
- Galvanic Skin Response: Electrical conductivity between two points on the wearer's arm
- Skin Temperature: Temperature of the skin and is generally reflective of the body's core temperature
- Near-Body Temperature: Air temperature immediately around the wearer's armband.

Experiments with ODT [H. Kargupta]

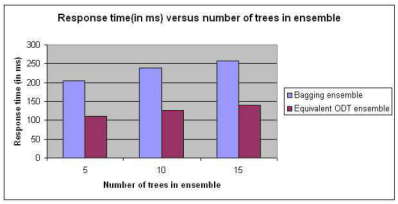
construction of ODTs using four C4.5 trees

reports the structure of an ODT obtained by projecting the trees along the first principle component.

aggregated orthogonal decision trees have accuracy comparable to that of large Bagging ensembles.

an aggregated ODT is a good solution for classification problems on PDAs, pocket PCs or cell-phones

Performance analysis in ODT [H. Kargupta]



ODT– resource constraint [H. Kargupta]

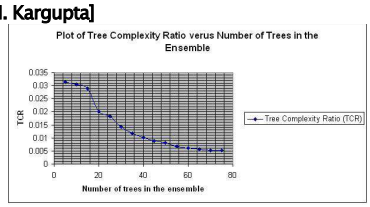
In resource constrained environments it is often necessary to keep track of the amount of memory used to store the ensemble.

The current implementation storing a node data structure in a tree requires approximately 1 KB of memory.

Consider an ensemble of 20 trees. If the average number of nodes in the trees in the Bagging ensemble is 7, then we are required to store 140 KB of data.

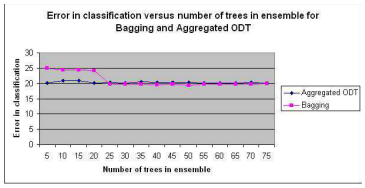
Orthogonal decision trees on the other hand are smaller in size, with less redundancy. In the experiments they typically have a complexity of 3 nodes. This means that to store only 60 KB of Data are required

TCR [H. Kargupta]



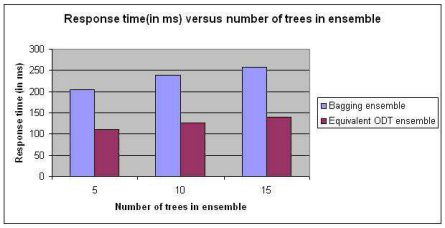
•Tree Complexity Ratio (TCR): total number of nodes in the ODT versus the total number of nodes in the Bagging ensemble
•It may be noted that in resource constrained environments one can opt for meaningful trees of smaller size and comparable accuracy as opposed to larger ensembles with a slightly better accuracy

ODT error in classification [H. Kargupta]



Response time comparison [H. Kargupta]

on a pocket PC using a Bagging ensemble and an equivalent orthogonal decision tree ensemble



Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

Block 5: Mining under resource constraints

- Introduction (Ernestina Menasalvas)
- Approaches: Classification, Clustering and Association

Block 6: Conclusions and Outlook

Gaber– AOG (Algorithm Output Granularity)

Adapting the algorithm output according to:

- resource availability and
- data stream generation/input rate.

The AOG approach is based on the following axioms:

1.The algorithm output rate (AR) is function in a data rate (DR),

- AR = f(DR).

2.The time needed to fill the available memory by the algorithm results (TM) is function in (AR)

- TM = f(AR).

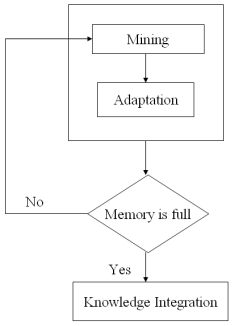
3.The algorithm accuracy (AC) is function in (TM)

- AC = f(TM).

Gaber–AOG (cont)

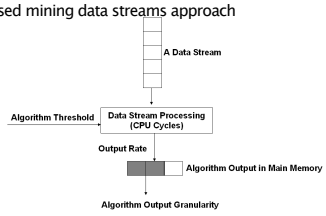
Main idea:
change the threshold value that in turn changes the algorithm rate according to three factors:
1.History of data rate to algorithm rate ratio.
2.Remaining time.
3.Remaining memory.

Target:
to keep the balance between the algorithm rate and data rate from one side and the remaining time and remaining memory from the other side.



Gaber–AOG (cont.)

- three–stage,
- resource–aware
- distance–based mining data streams approach



The process of mining the data stream (Gaber)

1. Determine the frequency of adaptation and mining.
2. According to the data rate, calculate the algorithm output rate and the algorithm threshold.
3. Mine the incoming stream using the calculated algorithm threshold.
4. Adjust the threshold after a time frame to adapt with the change in the data rate using linear regression
5. Repeat steps 3 and 4 till the algorithm lasts the time interval threshold.
6. Perform knowledge integration of the results

AOG based algorithms

- LightWeight Clustering (LWC): the threshold is used to specify the minimum distance between the cluster center and the data element/record;
- LightWeight Classification (LWClass): In addition of using the threshold in specifying the distance, the class label is checked. If the class label of the stored items and the new item that are similar (within the accepted distance) is the same, the weight of the stored item is increased along with the weighted average of the other attributes, otherwise the weight is decreased and the new item is ignored;
- LightWeight Frequent patterns (LWF): the threshold is used to determine the number of counters for the heavy hitters.

LWC[gaber]

```
1.  x = 1, c = 1, M = number of memory blocks
    available
2.  Receive data item DI[x]
3.  Center[c] = DI[x]
4.  M = M - 1
5.  Repeat
6.      x = x + 1
7.      Receive DI[x]
8.      For i = 1 to c
9.          Measure the distance between Center[i]
            and DI[x]
10.     If distance > dist (the threshold)
11.     Then
12.         c = c + 1
13.         If (M <> 0)
14.         Then
15.             Center[c] = DI[x]
16.         Else
17.             Merge DI[]
18.     Else
19.         For j = 1 to c
20.             Compare between Center[j] and DI[x] to
                find shortest distance
21.             Increase weight for Center[j] by the
                shortest distance
22.             Center[j] = (Center[j] * weight + DI[x]) /
                (weight + 1)
24. Until Done.
```

incrementally add new data elements to existing clusters according to an adaptive threshold value

distance between the new data point and all existing cluster centers > than the current threshold value, then create a new cluster

Lwclass [gaber]

```
1. x = 1, c = 1, M = number of memory blocks available
2. Receive data item DI[x]
3. Center[c] = DI[x]
4. M = M - 1
5. Repeat
6.  x = x + 1
7.  Receive DI[x]
8.  For i = 1 to c
9.      Measure the distance between Center[i] and DI[x]
10. If distance > dist (The threshold)
11. Then
12.     c = c + 1
13.     If (M <> 0)
14.     Then
15.         Center[c] = DI[x]
16.     Else
17.         Merge DI[]
18. Else
19.     For j = 1 to c
20.         Compare between Center[j] and DI[x] to find
            shortest distance
21.     If CL[j] = CL[x]
22.     Then
23.         Increase weight for Center[j] with shortest
            distance
24.         Center[j] = (Center[j] * weight + DI[x]) /
            (weight + 1)
25.     Else
26.         Increase weight for Center[j] with shortest
            distance
27. Until Done
```

determining the number of instances according to the available space in the main memory

searches for the nearest instance already stored in the main memory according to a pre-specified distance threshold

If the class label is the same, it increases the weight for this instance by one

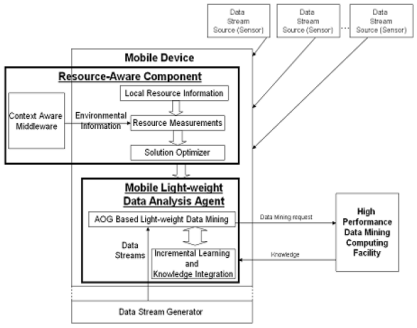
decrements the weight by one. If the weight becomes zero, this element will be released from the memory

LWF [gaber]

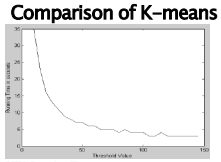
```
1. Set the number of top frequent items to k
2. Set the counter for each k
3. Repeat
4.  Receive the item
5.  If the item is new and one of the k counters are 0
6.  Then
7.      Put this item and increase the counter by 1
8.  Else
9.      If the item is already in one of the k counters
10.     Then
11.         Increase the counter by 1
12.     Else
13.         If the item is new and all the counters are full
14.         Then
15.             Check the time
16.             If time > Threshold Time
17.             Then
18.                 Re-set number of least n of k counters to 0
19.                 Put the new item and increase the counter by 1
20.             Else
21.                 Ignore the item
22.                 Decrease all the counters by 1
23. Until Done
```

number of freq. items according to the available memory

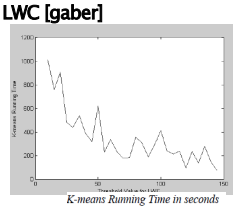
Resource aware architecture [gaber]



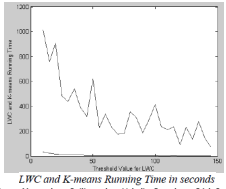
Comparison of K-means and LWC [gaber]



LWC Running Time



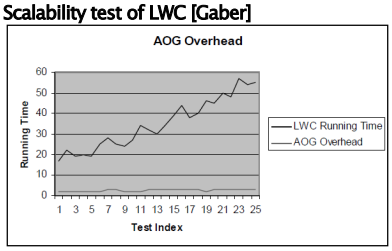
K-means Running Time in seconds



LWC and K-means Running Time in seconds

- LWC outperforms K-means even with the fine threshold that leads to creating large number of clusters
- The generated centers for both algorithms are very close and have the same trend

Scalability test of LWC [Gaber]



•the unpredictability of the number of passes needed by K-means leads to fluctuating running time with similar data set sizes
•Experiment with an increase in the dataset sizes to show stability in the AOG overhead.

Granularity-based Approach [Gaber]

Combining the three possible granularity-based adaptation, namely:

1. AIG: Algorithm Input Granularity
2. AOG: Algorithm Output Granularity
3. APG: Algorithm Processing Granularity

Granularity based algorithms [Gaber]

Clusterers:

- Light-Weight Clustering
- RA-Cluster
- DRA-Cluster
- RA-VFKM

Change Detection:

- CHANGE-DETECT

Classifiers:

- Light-Weight Class (LWClass)
- RA-Class
- DRA-Class

Time Series Analysers:

- RA-SAX

Frequent Items and Associations:

- LWF (Light-Weight Frequent Items)
- HiCoRE (Highly Correlated Energy-Efficient Rules)

Gaber. Resource aware-Clustering

enables resource-awareness in streaming computation

using algorithm granularity settings changes the resource consumption patterns periodically.

framework is applied to a novel threshold-based microclustering algorithm to test its validity and feasibility.

RA-Cluster.

- the first stream clustering algorithm that can adapt to the changing availability of different resources.
- The experimental results showed the applicability of the framework and the algorithm in terms of resource-awareness and accuracy

Algorithm settings

- Input granularity (AIG):
  - sampling, load shedding, and creating data synopsis techniques.
- Algorithm Output Granularity AOG :
  - number of knowledge structures created or level of output granularity.
- Algorithm Processing Granularity APG: changing the processing settings of the algorithm to consume smaller amount of resources according to consumption measures over the last frame :
  - error rate of approximation algorithms.

Algorithm settings :

Algorithm Input Granularity (AIG):

- Sampling: is the process of statistically choose some data records to be processed.
- Load shedding: represenst the process of dropping a chunk of data records from being processed. This could be an appropriate technique to stop the processing to enable some optimization process to be done during this time. It is considered to be a direct solution if there is a burst in data streams. We can shed the load and continue the processing after the burst.
- Creating data synopsis: summarizing or compressing the incoming data on the fly before it is being processed:
  - Wavelets
  - simple statistical summarization techniques

Algorithm settings (II)

Algorithm Output Granularity (AOG):

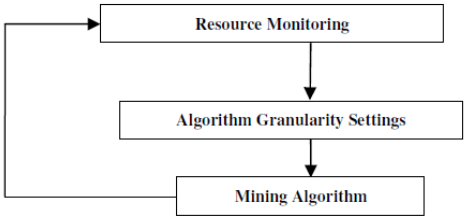
- number of knowledge structures:
  - number of clusters or rules.
- The output size could be changed also using level of output granularity which means the less detailed output, the higher the granularity and vice versa.

Algorithm Processing Granularity (APG):

- Randomization and approximation techniques

The process of enabling resource awareness should be very lightweight in order to be feasible in a streaming environment characterized by its scarcity of resources

System architecture [Gaber]



RA–cluster algorithm

RA–Cluster:combines resource–awareness, adaptation and real–time all in a holistic approach.

Process:

1. starts with using an initial threshold to run the algorithm and after a fixed time frame the resource consumption patterns of the CPU, memory, and battery given that we run in a resource constrained environment is assessed .

2.According to the above assessment, the algorithm settings are changed to cope with the data rate.

RA–Cluster is an incremental online micro–clustering algorithm that has all the required parameters to enable resource–awareness


```
Repeat
Repeat
Get next DS record DSRec
Find ShortDist which is the shortest distance between DSRec
and micro-cluster centers
If ShortDist < Radiusthreshold
Assign DS record to that micro-cluster
Update micro-cluster statistics
Else
Create new micro-cluster
End
Until (END-OF-TIME-FRAME)
Calculate NoFMem, NoFCPU, NoFBatt
If NoFMem < RTMem
Reclaim outlier memory
Increase Radiusthreshold (discourage micro-luster creation)
ElseIf if available memory increases
Decrease Radiusthreshold (encourage micro-cluster creation)
End
If NoFCPU < RTCPU
Decrease randomization factor (less processing per unit)
ElseIf unused CPU power increases
Increase randomization factor (more processing per unit)
End
If NoFBatt < RTBatt
Decrease sampling rate (slower consumption pattern)
ElseIf remaining battery life increases
Increase sampling rate (faster consumption pattern)
End
Until (END-OF-STREAM)
```

[Gaber]

Some results [Gaber]

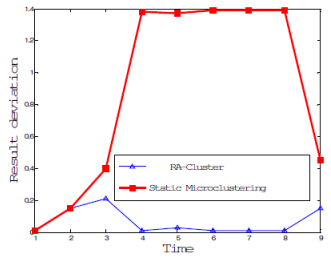


Figure 10. RA-Cluster over Evolving Data Stream

AG main lacks [Gaber]

AG model has proven its applicability to change the resource consumption, but:

- The bounds over the AG settings have no guarantee over the quality of the output. quality relies on other interleaving factors such as data distribution and the running mining technique.
- changes in the AG settings are not quality–aware. the algorithm changes according only to the availability of computational resources.
- loss in accuracy :in some cases, we can gain the same accuracy using less resources.
- The AG settings do not take into consideration the interaction among the different settings. Addressing this issue can optimize the use of resources.

Quality assurance (Gaber, Franke, Karnstedt)

Generic 3 layer model for quality guaranteed resource-aware (QGRA) data mining on data streams.

applicable to a wide variety of stream mining techniques.

The model bridges the gap between quality aware mining and general resource-adaptivity by monitoring the resource consumption.

Quality model

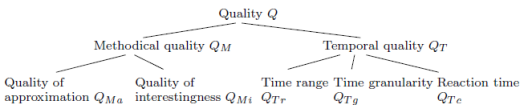
To control the adaptation the algorithm uses:

1.Function to determine the algorithmic parameters from the assessed resources.

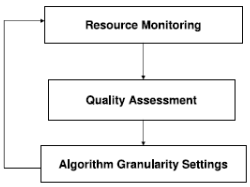
2.Function to determine the output quality based on the chosen algorithmic parameters.

3.Functions to compute lower bounds for the algorithmic parameters in order to maintain the quality of the output.

Different quality measures [Gaber, Franke, Karnstedt]



The three layers [Gaber, Franke, Karnstedt]



1. resource monitoring: works over dynamic time intervals. dynamic time window that changes according to the criticality of the available computational resources.
2. real-time quality assessment. Able to provide information about the quality of the output given the availability of resources. Also provides the system with information about preserving computational resources while maintaining the same quality
3. (AGS) component:feeds the mining algorithms with input, output and processing settings

QRSA algorithm

1: $R_t = \text{res-mon}(t)$
2: **if** $0 \neq \rho(R_L, \cup_{i=1}^{t} R_i, P_t)$ **then**
3: $P = \tau(R_L, \cup_{i=1}^{t} R_i, P_t)$
4: $P' = \omega(Q_L, \cup_{i=1}^{t} Q_i, P_t, P$
5: **if** $\Phi(R_L, \psi(P'))$ **then**
6: $P = P'$
7: **end if**
8: **if** $!(\Phi(R_L, \phi(P)) \wedge \Psi(Q_L, \psi(P)))$ **then**
9: **return** false
10: **end if**
11: **else**
12: $P = P_t$
13: **end if**
14: $Q_{t+1} = \{q, x, t+1 | (q, x) \in \psi(P) \wedge \bar{B}(u, y)$
15: $P_{t+1} = \{p, x, t+1 | (p, x) \in P \wedge \bar{B}(o, y) \neq (p$
16: set parameters P
17: **return** true

- resource monitoring component
- dynamically provided resource limits,
- If any adaptation is necessary, a new set of parameters is determined refine these parameters.
- analyze the set of parameters suggested by the resource adaptation as additional input.
- Only if the parameters modified like this still meet the resource limits they are

- parameters are checked again.

- the resulting qualities and parameters are stored

- if no adaptation takes place, parameters for time t +1 are set to those from time t in order to build the timelined sets (line 12).

Requirements of the algorithms

- parameters must exist in the algorithm. a strong correlation between the adaption factors and the algorithm's resource requirements helps estimating the quality of the output.
- the algorithm must show homogeneous behavior
- stream with homogeneus properties maintained
- Able to establish a partitioning into independent sections in the mining result. Thus different values of the adaption factors only have "local" effects in the mining results.
- possible to query each of these independent sections separately.

End of Block 5

Thank you!

Questions?

References (Block 5) (1/5)

Marcel Karnstedt, Conny Franke, Mohamed Medhat Gaber: A Model for Quality Guaranteed Resource-Aware Stream Mining, Fifth International Workshop on Knowledge Discovery from Ubiquitous Data Streams icw ECML/PKDD'07, Warsaw, Poland, Sep 2007, pp. 72-82

Conny Franke, Marcel Karnstedt, Kai-Uwe Sattler: Mining Data Streams under Dynamicly Changing Resource Constraints, KDML 2006: Knowledge Discovery, Data Mining, and Machine Learning, October 2006, Hildesheim, Germany, pp. 262-269

Conny Franke, Michael Hartung, Marcel Karnstedt, Kai-Uwe Sattler: Quality-aware Mining of Data Streams, <international Conference on Information Quality (ICIQ) 2005, Boston, USA

References (Block 5) (2/5)

Haghighi P. D., Krishnaswamy S., Zaslavsky A., Gaber M. M., and Loke S. W., Context-Aware Adaptive Data Stream Mining, in Gama J., Ganguly A., Omitaomu O., Vatsavai R. and Gaber M. M. (Eds), Intelligent Data Analysis, Special Issue on Knowledge Discovery from Data Streams Volume 13, Number 3, 2009, pp. 423-434, IOS Press, ISSN 1088-467X (Print) 1571-4128 (Online).

Gaber M. M., and Yu P. S., A Holistic Approach for Resource-aware Adaptive Data Stream Mining, Journal of New Generation Computing, ISSN 0288-3635 (Print) 1882-7055 (Online), Volume 25, Number 1, November, 2006, pp. 95-115, Ohmsha, Ltd., and Springer Verlag.

Gaber M. M., and Yu P. S., Detection and Classification of Changes in Evolving Data Streams, International Journal of Information Technology & Decision Making, Vol. 5, No. 4, pp. 659-670, World Scientific Publishing Company, 2006.

References (Block 5) (3/5)

Krishnaswamy S., Gaber M. M., Harbach M., Hugues C., Sinha A., Gillick B., Haghighi P. D., and Zaslavsky A., Open Mobile Miner: A Toolkit for Mobile Data Stream Mining, Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2009, June 28 – 1 July, Paris, France. (Demo paper).

Haghighi P. D., Gaber M. M., Krishnaswamy S., Zaslavsky A, and Loke S. W., An Architecture for Context-Aware Adaptive Data Stream Mining, Proceedings of the International Workshop on Knowledge Discovery from Ubiquitous Data Streams (IWKDUDS07), in conjunction with ECML and PKDD 2007, September 17, Warsaw, Poland, 2007

Phung N. D., Gaber M. M., and Röhm U., Resource-aware Distributed Online Data Mining for Wireless Sensor Networks, Proceedings of the International Workshop on Knowledge Discovery from Ubiquitous Data Streams (IWKDUDS07), in conjunction with ECML and PKDD 2007, September 17, Warsaw, Poland, 2007

References (Block 5) (4/5)

Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A., Resource-Aware Mining of Data Streams, Journal of Universal Computer Science, Vol. 11, No. 8 (2005), pp. 1440-1453, ISSN 0948-695x, Special Issue on Knowledge Discovery in Data Streams, Jesus S. Aguilar-Ruiz and Joao Gama (Eds.), Verlag der Technischen Universität Graz, Know-Center Graz, Austria, August 2005.

Hamed Chok, Le Gruenwald: Spatio-temporal association rule mining framework for real-time sensor network applications. CIKM 2009: 1761-1764

Nan Jiang, Le Gruenwald: CFI-Stream: mining closed frequent itemsets in data streams. KDD 2006: 592-597

Wei-Guang Teng, Ming-Syan Chen, and Philip S. Yu; Resource-Aware Mining with Variable Granularities in Data Streams; SIAM Int'l Conf. on Data Mining; 2004

References (Block 5) (5/5)

M. El-Hajj and O.R. Zaiane. COFI-tree mining: a new approach to pattern growth with reduced candidacy generation. In Proc. FIMI 2003.

Two Mining Data Streams under Dynamicly Changing Resource Constraints. Conny Franke Marcel Karnstedt Kai-Uwe Sattler

W. Teng, M. Chen, and P. S. Yu, Resource-Aware Mining with Variable Granularities in Data Streams, Proc. Of SIAM SDM 2004.

Y. Chi, P. S. Yu, H. Wang, R. R. Muntz, Loadstar: A Load Shedding Scheme for Classifying Data Streams, Proc. Of SIAM SDM 2005.

Presentation Outline

☑Block 1: Introduction

☑Block 2: Supervised learning on streams

☑Block 3: Unsupervised learning on streams

☑Block 4: Mining evolving social data

☑Block 5: Mining under resource constraints

Block 6: Conclusions and Outlook

Outlook I

Learning on complex stream data

- multiple, interrelated and interdependent streams
- Data that involve static objects & streams feeding them
- Probabilistic models and tensor-based learning
- Object profiling:
  A model for all data or a model for each object (or both) ?

Multiple sensors Communities

Customers, Users, Patients

Challenges of new applications

- Data from mobile devices
- Data from devices that are available irregularly
- Learning using limited computational resources
- Learning under time constraints
- Capturing context

Outlook II – Old challenges, not yet solved

- Dealing with time: *Multi-horizon* and *multi-granularity* analysis
- Scalability for large data volumes
  - Decrease complexity
  - Increase parallelism

  MapReduce Clouds
- Robustness, esp. if the learners are complex
- Learning for online or realtime applications
- Visualization

  Visualization of WHAT ?
  The streams, the objects, the models

  Visualization for WHOM?
  The expert, the data owner, the decision-maker, the casual observer
- Evaluation: Measures & Benchmarks

Outlook II – Old challenges,not yet solved

Supervised learning for evolving data

- Dealing with emerging and evolving concepts
- Delayed labeling
- Label availability
- Cost-benefit trade-off of the model update
- Change description
- Predicting re-occurring contexts
- Multi-label prediction
- Reliability and uncertainty

Outlook II – Old challenges,not yet solved

Unsupervised learning for evolving data

- Dealing with emerging and evolving concepts
- Waiving the assumptions about the data generating process (or verifying them)
- Change description
- Setting up the learner:
  - What is the influence of parameter x, given that there is no ground truth in the data?
  - If Gibbs sampling, then on what sample?
- Evaluation

Outlook – Challenges, amplified by evolving *social* data

Community identification in evolving social data

- Exploitation of evolving communities

  Event prediction

  Opinion mining

  Trend identification
- Visualization of community evolution
  - Human-friendly
  - Demonstrating the role of communities and the impact of change in a comprehensible way
  - Application-specific
- Supporting online applications

  Recommendation engines

Thank you very much!

QUESTIONS