Philipp Zormeier

Event Detection in Social Streams

14.11.2012
„Mining Volatile Data“

Outline

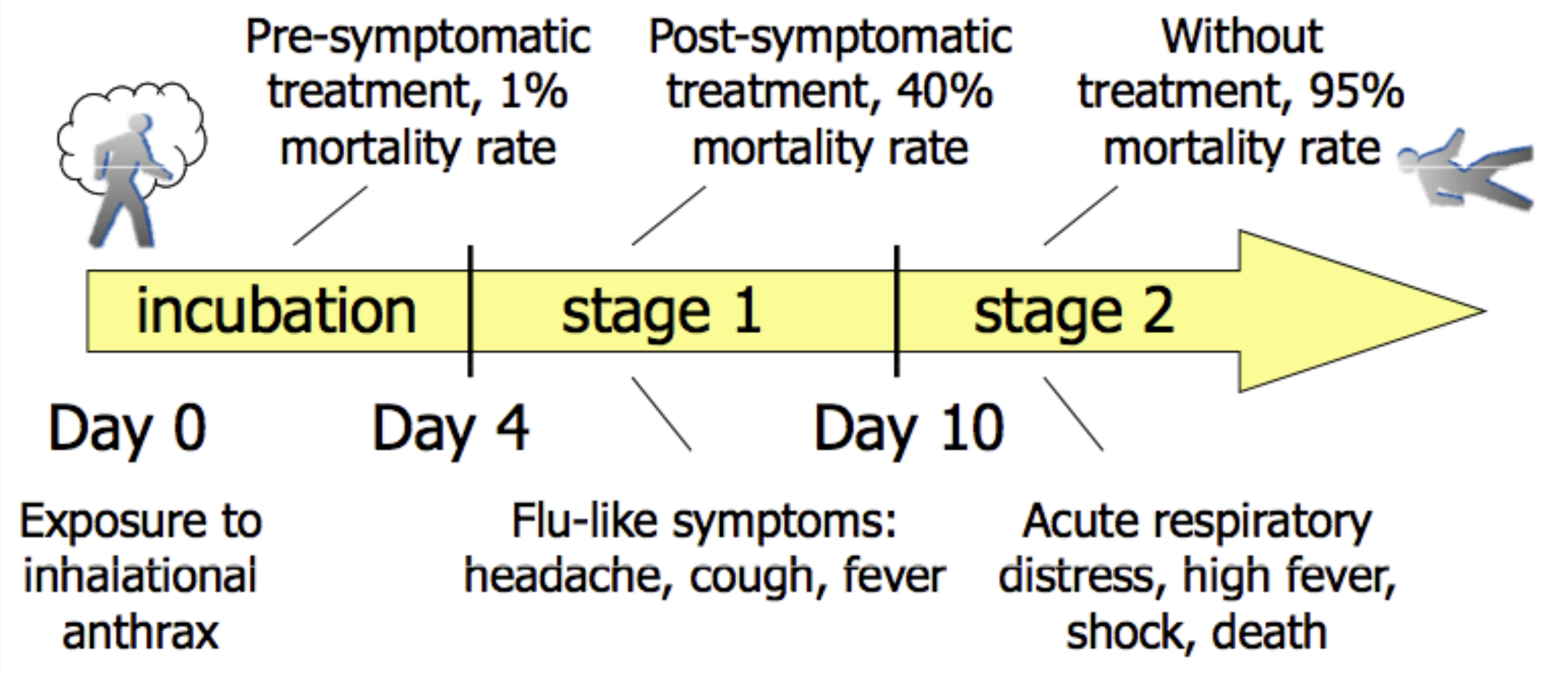
What is Event Detection?

- Analysis of monitoring data
- Detection of interesting changes
- Characterization of the event

I. Introduction

Applications of Event Detection?

Famous example:



I. Introduction

Social Streams

- People post about their lives
- Important events are captured in bursts of posts
- Continuous interaction
- Posts contain **temporal**, **structural** and **content** information

I. Introduction

Key Challenges

(i) Ability to use both the content and the structure of the interactions for detection

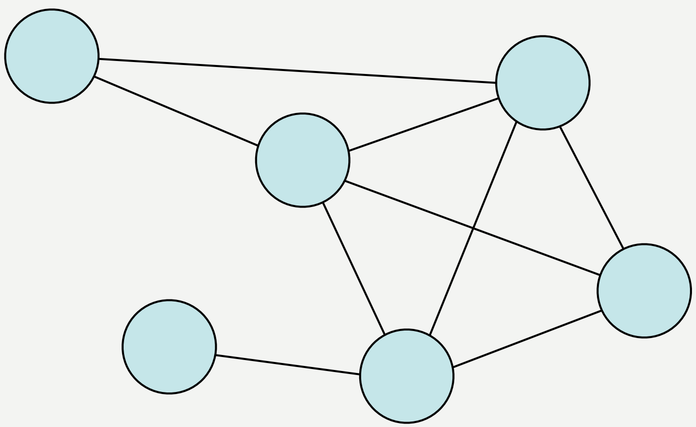
(ii) Ability to use temporal information

(iii) Ability to handle very large and massive volumes of text documents under the one-pass constraint

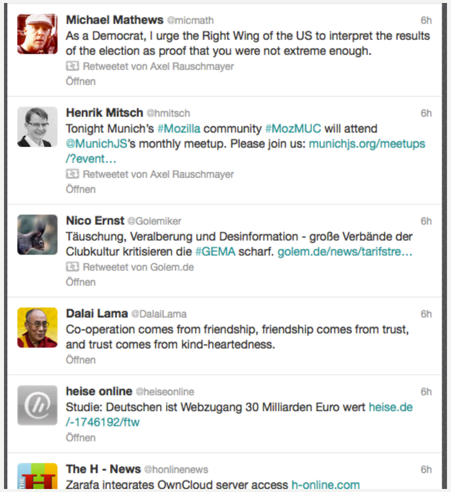
II. Social stream model

What do we call a Social Stream?

Social Stream = Structure + Content



+



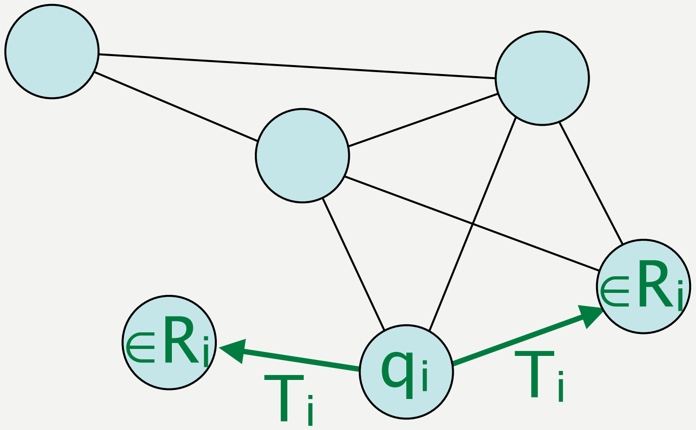
Definition of a Social Stream

Continuous and temporal Sequence of objects S_1 .. S_r ..

such that each $S_i = (q_i, R_i, T_i)$ contains..

- a text document T_i
- an origination node $q_i \in N$
- a set of receiver nodes

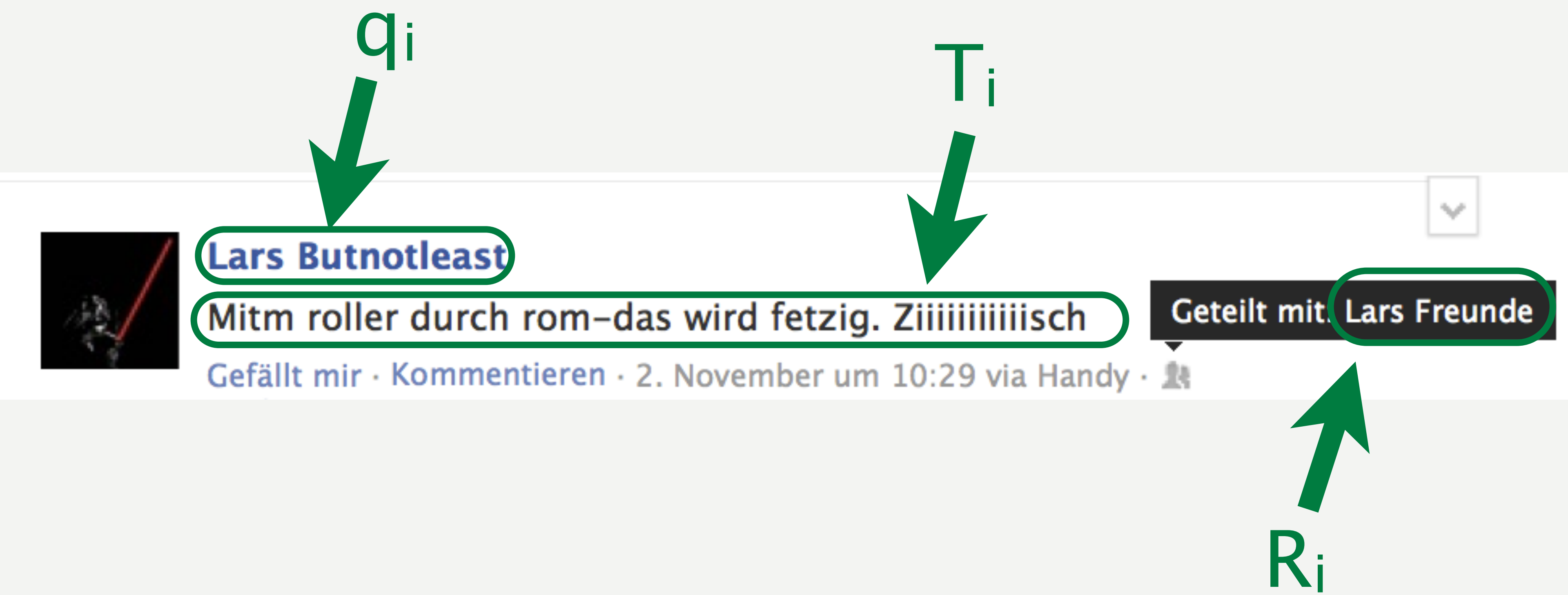
  $R_i \subseteq N$ ($\forall\, r \in R_i\ (q_i, r) \in A$)



Graph $G = (N,A)$

Examples: Facebook

Examples: Facebook



Examples: Twitter

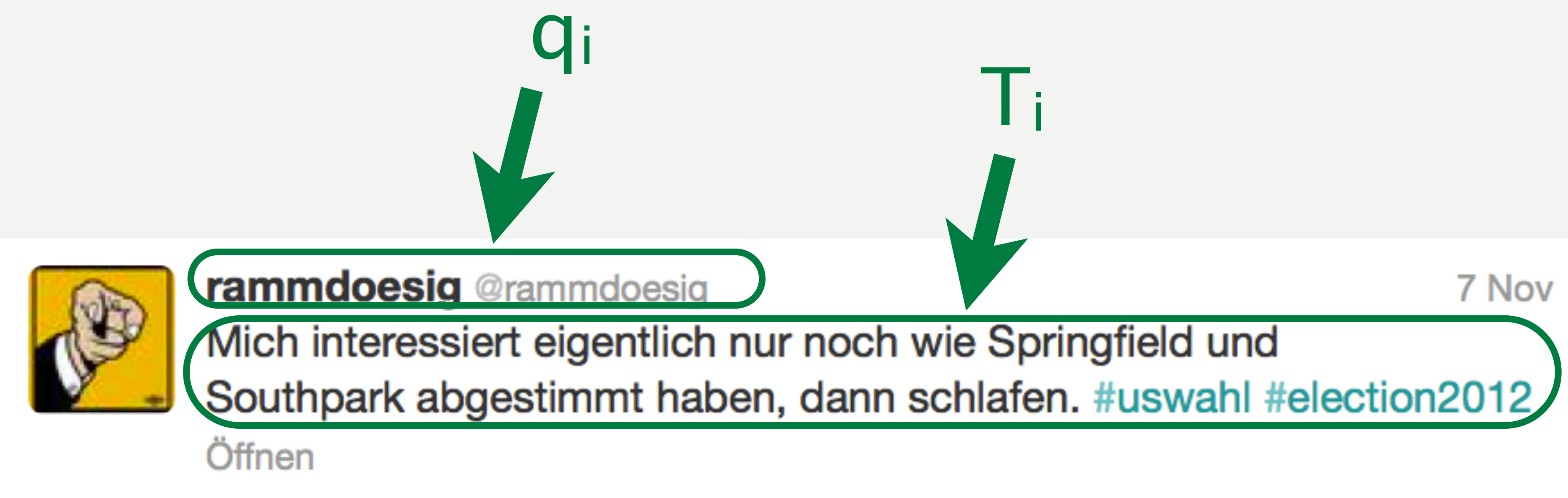
rammdoesig @rammdoesig 7 Nov

Mich interessiert eigentlich nur noch wie Springfield und Southpark abgestimmt haben, dann schlafen. #uswahl #election2012

Öffnen

II. Social stream model

Examples: Twitter



R_i not visible here (all followers)

Examples: Email

Example

Von: Philipp Zormeier

An: spam@siebenfarben.de

Hi,

This email is just an example.

Bye,

Philipp

Examples: Email

Data organization

Data points in Clusters

Cluster ~ Topics

New Point in Cluster ~ The actor talks about the topic

Many new points in Cluster ~ Something happened?

New Cluster ~ A new topic comes up

Social stream clustering

Partitioning the stream objects S_1 .. S_r ..
into k clusters C_1 .. C_k, such that

- for all i: S_i belongs to at most one cluster
- the similarity function uses content and network structure

Cluster summaries

$\Psi(C_i) = (V_i, \eta_i, W_i, \Phi_i)$

$V_i = \{j_{i1}, .. , j_{is}\}$: Set of nodes

$\eta_i = \nu_{i1} .. \nu_{is}$: Node frequencies

$W_i = \{l_{i1}, .. , l_{iu}\}$: Set of words

$\Phi_i = \varphi_{i1} .. \varphi_{iu}$: Word frequencies

Similarity of stream objects to clusters

Overall similarity:

$$\mathrm{Sim}(S_i, C_r) = \lambda \cdot \mathrm{SimS}(S_i, C_r) + (1 - \lambda)\, \mathrm{SimC}(S_i, C_r)$$

$\lambda \in [0,1]$

Similarity of stream objects to clusters

Structure:

$$\mathrm{SimS}(S_i, C_r) = \frac{\sum_{t=1}^{s_r} b_t \cdot v_{rt}}{\sqrt{\left\| R_i \cup \{q_i\} \right\| \cdot \left(\sum_{t=1}^{s_r} v_{rt} \right)}}$$

b binary vector: $b_t = 1$, if node $j_{rt} \in R_i \cup \{q_i\}$.

Similarity of stream objects to clusters

Content: $SimC(S_i, C_r)$ = TF–IDF

Similarity of stream objects to clusters

Overall similarity:

$$Sim(S_i, C_r) = \lambda \cdot SimS(S_i, C_r) + (1 - \lambda)\, SimC(S_i, C_r)$$

$\lambda \in [0,1]$

Assignment to clusters

New object

- » Find closest cluster
- » Similar enough? ($Sim(S_i, C_r) > \mu - 3 \cdot \sigma$?)
  - » no → create new cluster
  - » yes → assign to cluster

Case 1: Creation of clusters

A new data point is placed in its own new cluster.

The **most stale** cluster is replaced.

most stale = last recent updated

This is a **novel event.**

Case 2: Assignment to clusters

The data point is assigned to an existing cluster.

- Update cluster summary
- Check, if **evolution event** occured:

  $\Rightarrow$ Calculate fractional cluster presence

Case 2: Assignment to clusters

Fractional cluster presence F(t – H, t, C_i)

- Data arrival ratio in time period (t – H, t)
- Time horizon H has to be chosen

Evolution event:

$$\frac{F(t_c - H, t_c, C_i)}{F(t(C_i), t_c - H, C_i)} \geq \alpha$$

threshold α

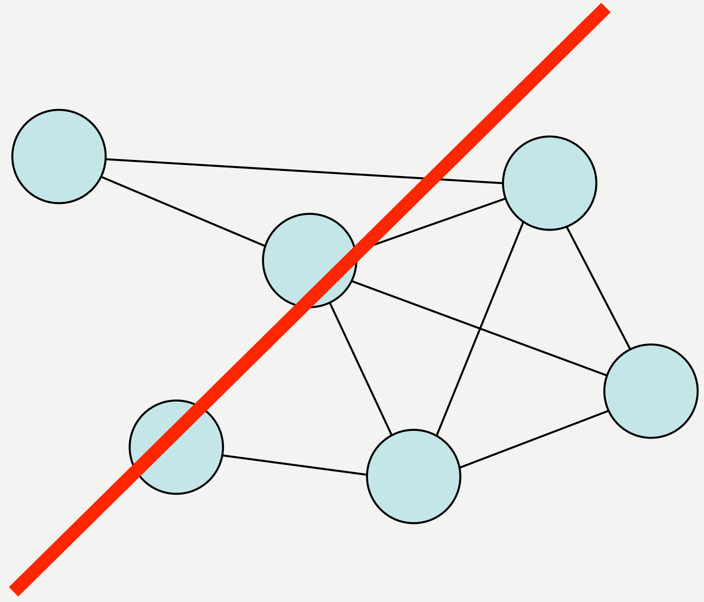
III. Unsupervised appr.

```
while(end of stream not reached)
   i = i + 1;
   Receive next object Si;
   for each cluster Cl compute Sim(Si,Cl);
   Let r be index of most similar cluster Cr;
   if(Sim(Si,Cr) < μ - 3·σ)
      then replace most stale cluster;
      else add Si to Cr and update Ψ(Cr);
   Update μ and σ;
```

III. Unsupervised appr.

```
while(end of stream not reached)
   i = i + 1;
   Receive next object Si;
   for each cluster Cl compute Sim(Si,Cl);
   Let r be index of most similar cluster Cr;
   if(Sim(Si,Cr) < μ - 3·σ)
      then replace most stale cluster;
            ⇒ novel event
      else add Si to Cr and update Ψ(Cr);
            ⇒ check for evolution event
   Update μ and σ;
```

Performance Issues



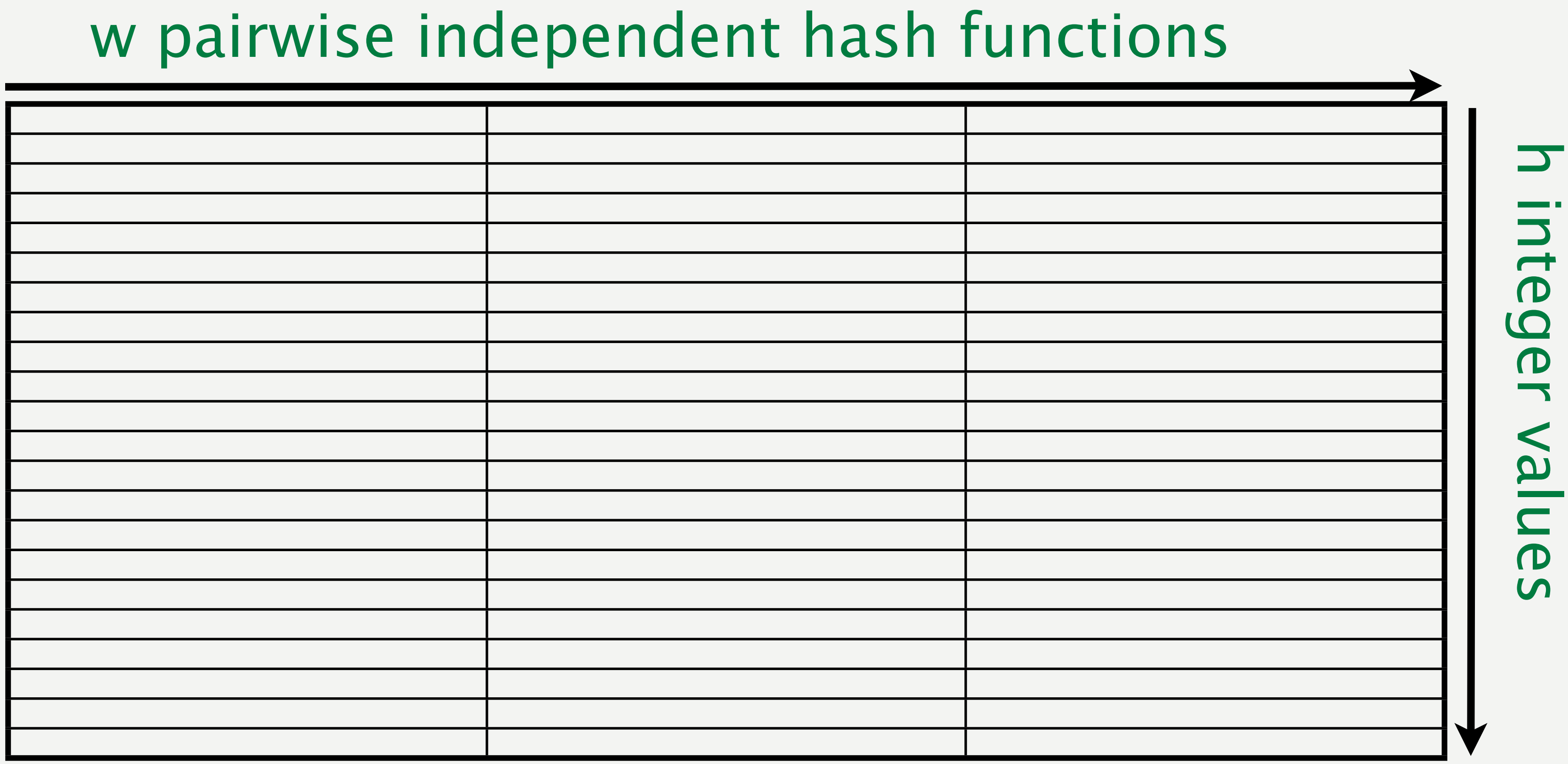
In reality it is a bit more complex..

II. Social stream model

Sketch-based Speedup

- Speed up node counting
- Using Count-min sketch (Hash based)
- Main idea: Estimate counts

Sketch-based Speedup



Supervised Event Detection

Assumptions:

- ✓ Known event E
- ✓ Access to history of the stream
- ✓ Information about set of relevant posts

Variation of Social Stream Clustering

Cluster replacement is not allowed

- Event Signature V(E)
  - Vector of relative distribution of event-specific stream objects to clusters
- Horizon Signature over $(t_c - H, t_c)$
  - Vector of relative distribution of arriving points

Supervised Event Detection

✓ Calculate dot product of horizon signature and event signature

✓ Events are signaled at certain alarm level

Data sets

1) **Twitter Social Stream**

1,628,779 tweets

59,192,401 nodes (avg ~84 nodes/tweet)

2) Enron Email Stream (filtered)

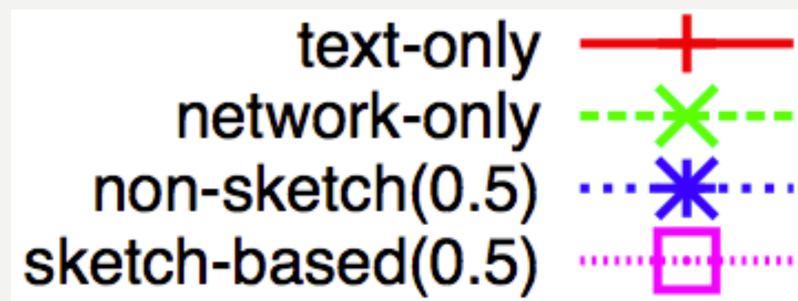
349,911 emails

29,083 nodes (avg ~3.62 receivers/email)

Clustering performance

$$Sim(S_i, C_r) = \lambda \cdot SimS(S_i, C_r) + (1 - \lambda)\, SimC(S_i, C_r)$$

- ✓ λ = 0 text–only
- ✓ λ = 1 network–only
- ✓ λ = 0.5 non–sketch
- ✓ λ = 0.5 sketch–based



sketch table: h = 262,213, w = 2

V. Performance eval.

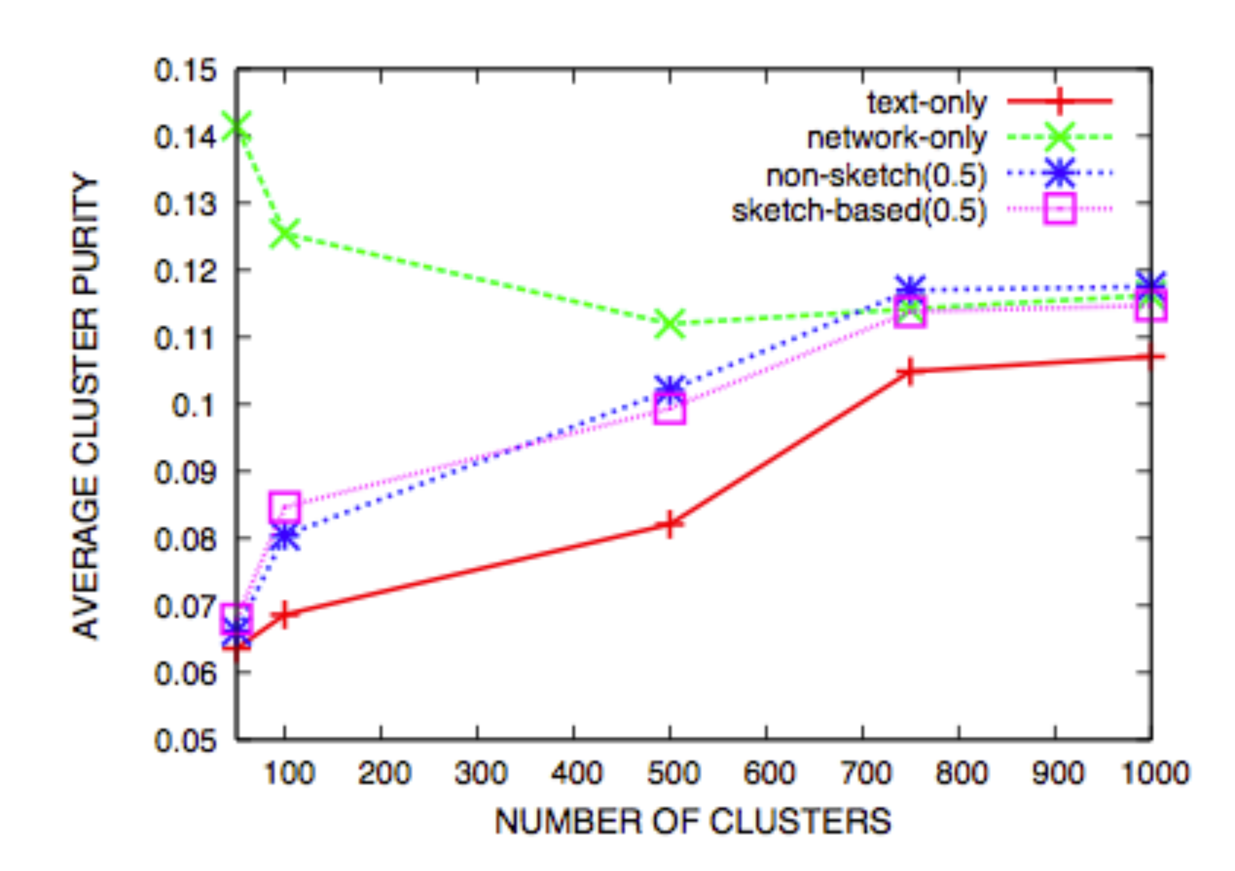
Effectiveness of the clustering

- Assumption: Frequent hash tag -> meaningful event
- Test for purity of clusters on dominant hash tags
- Purity = Fraction of objects with dominant tag

Efficiency of the clustering

- Number of stream objects processed by time

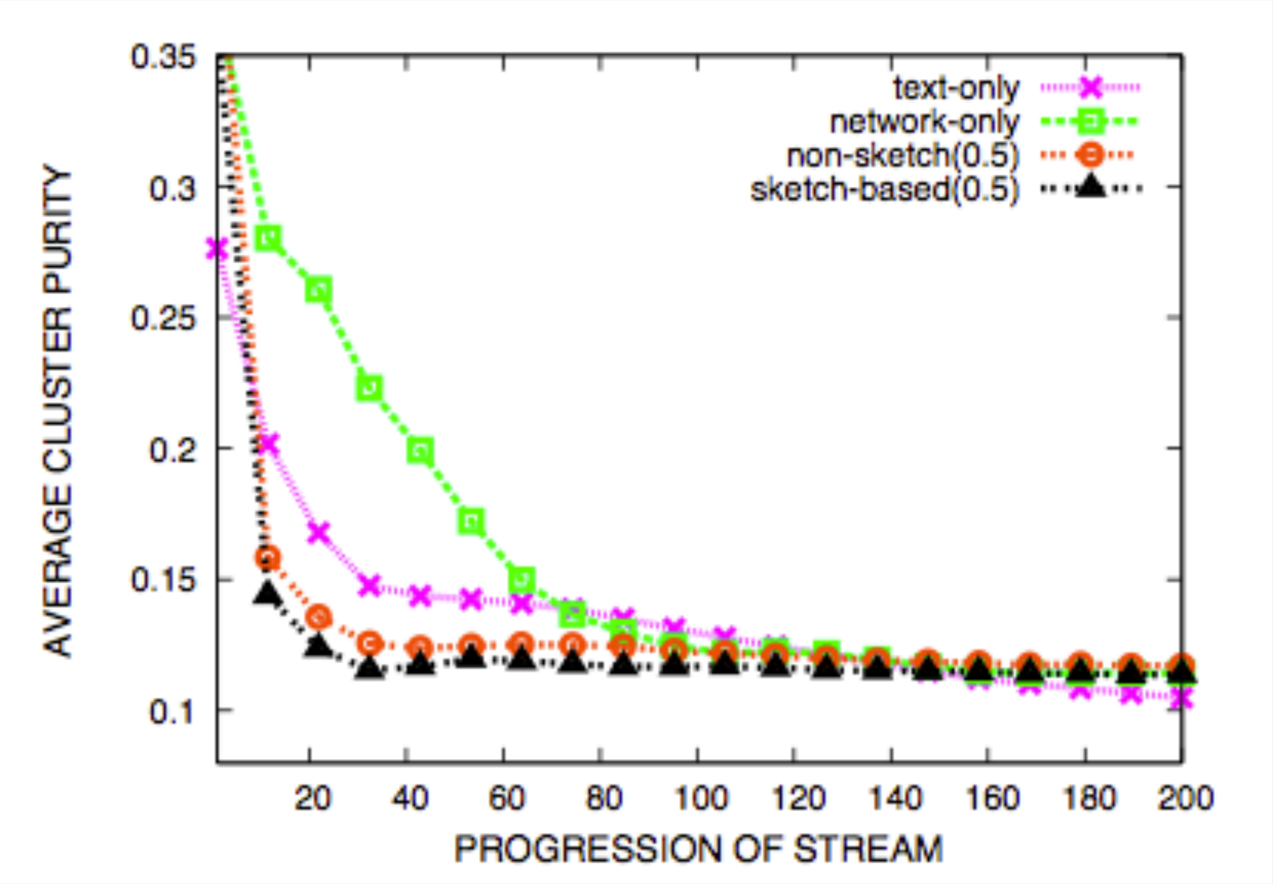
Effectiveness of the clustering



sketch
w = 262,213
h = 2

V. Performance eval.

Effectiveness of the clustering

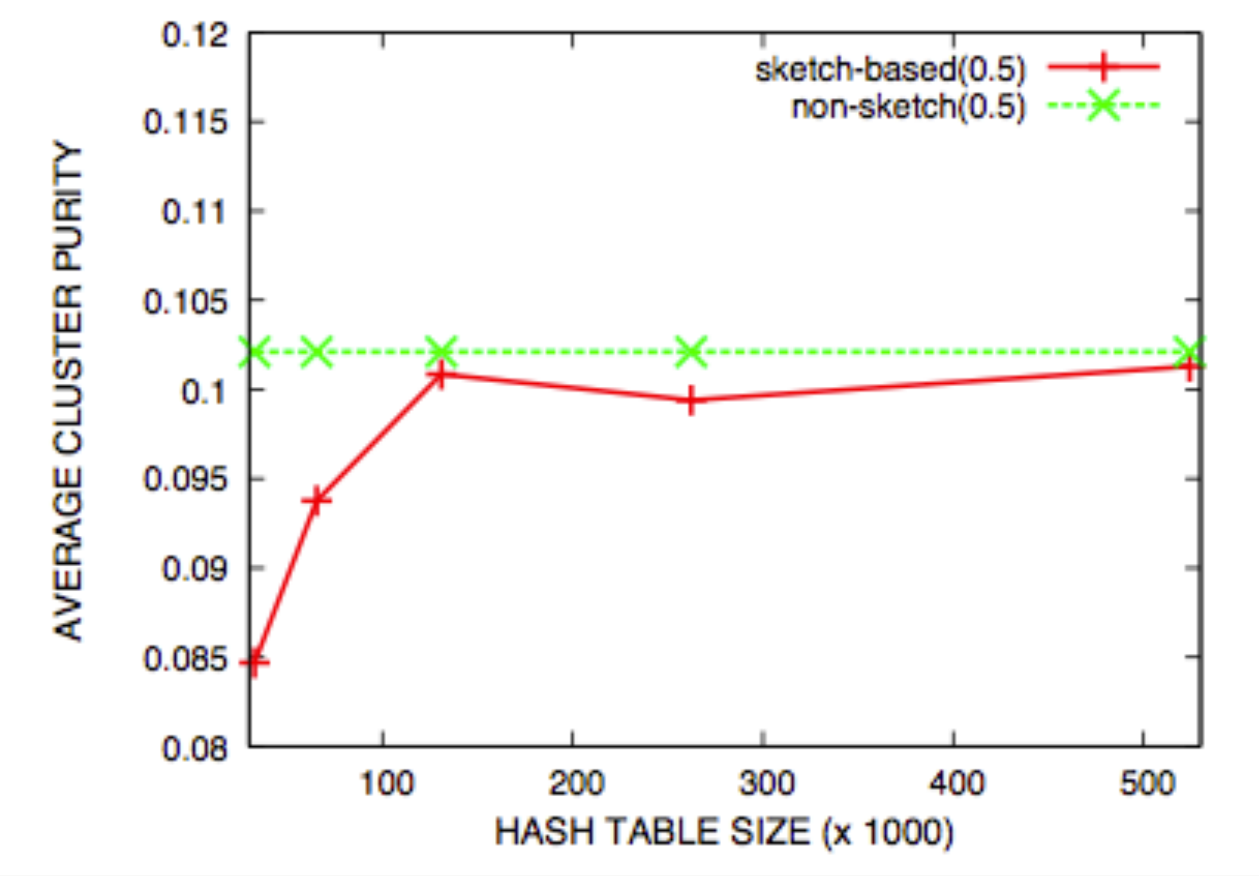


clusters
k = 750

sketch
w = 262,213
h = 2

V. Performance eval.

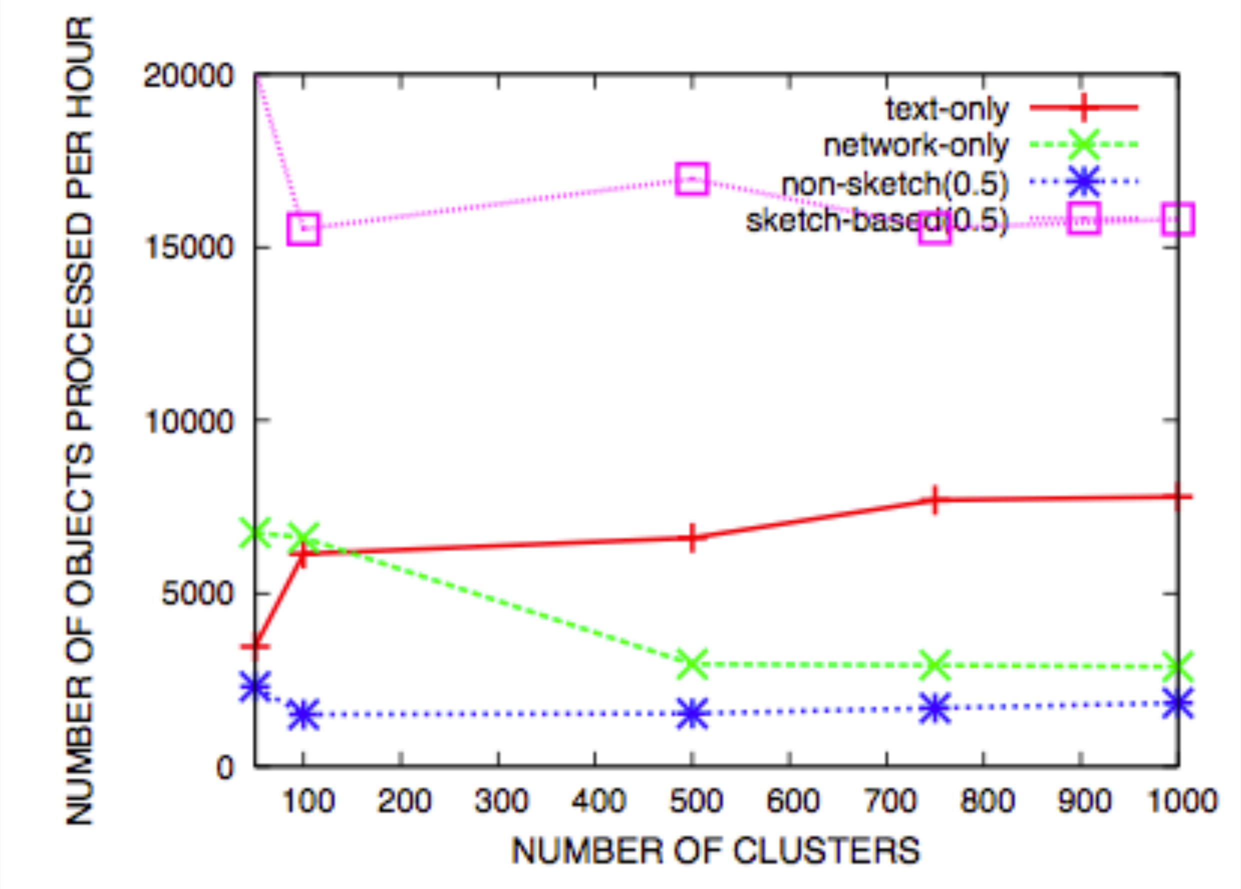
Effectiveness of the clustering



clusters
k = 500

V. Performance eval.

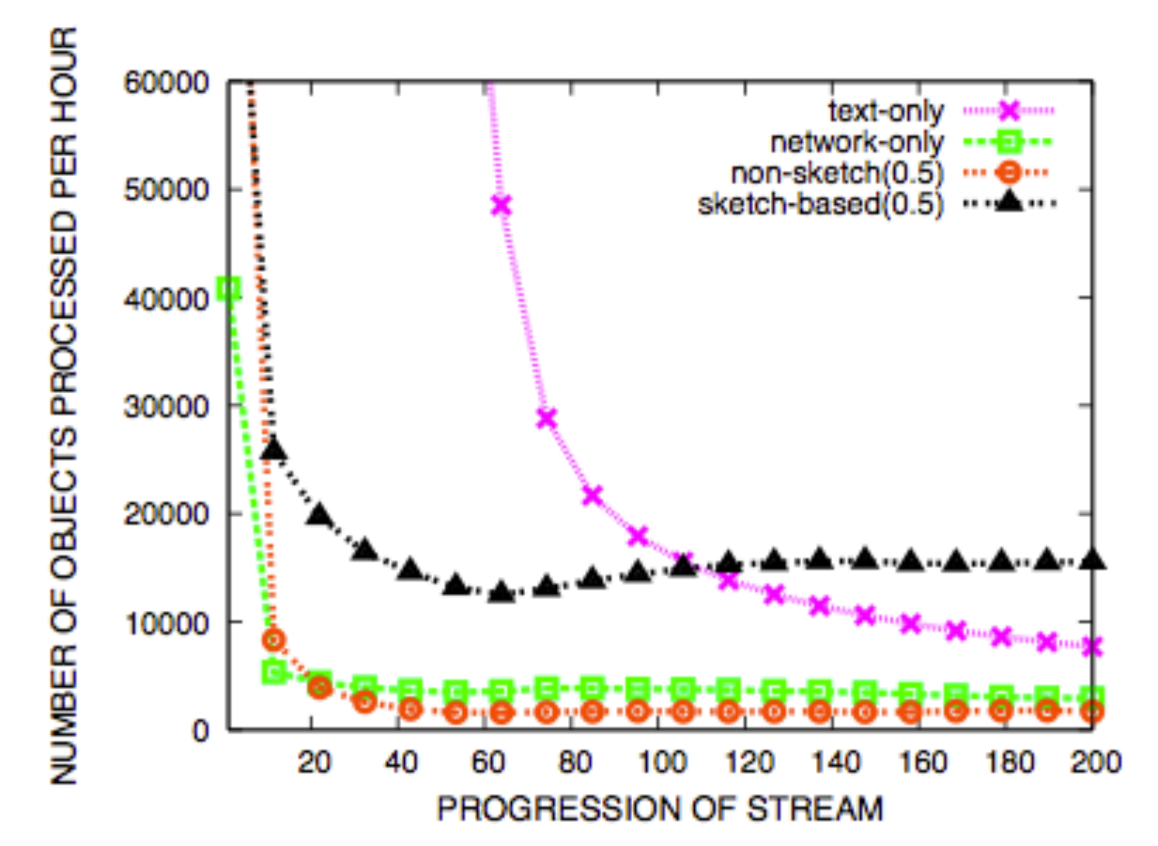
Efficiency of the clustering



sketch
w = 262,213
h = 2

V. Performance eval.

Efficiency of the clustering

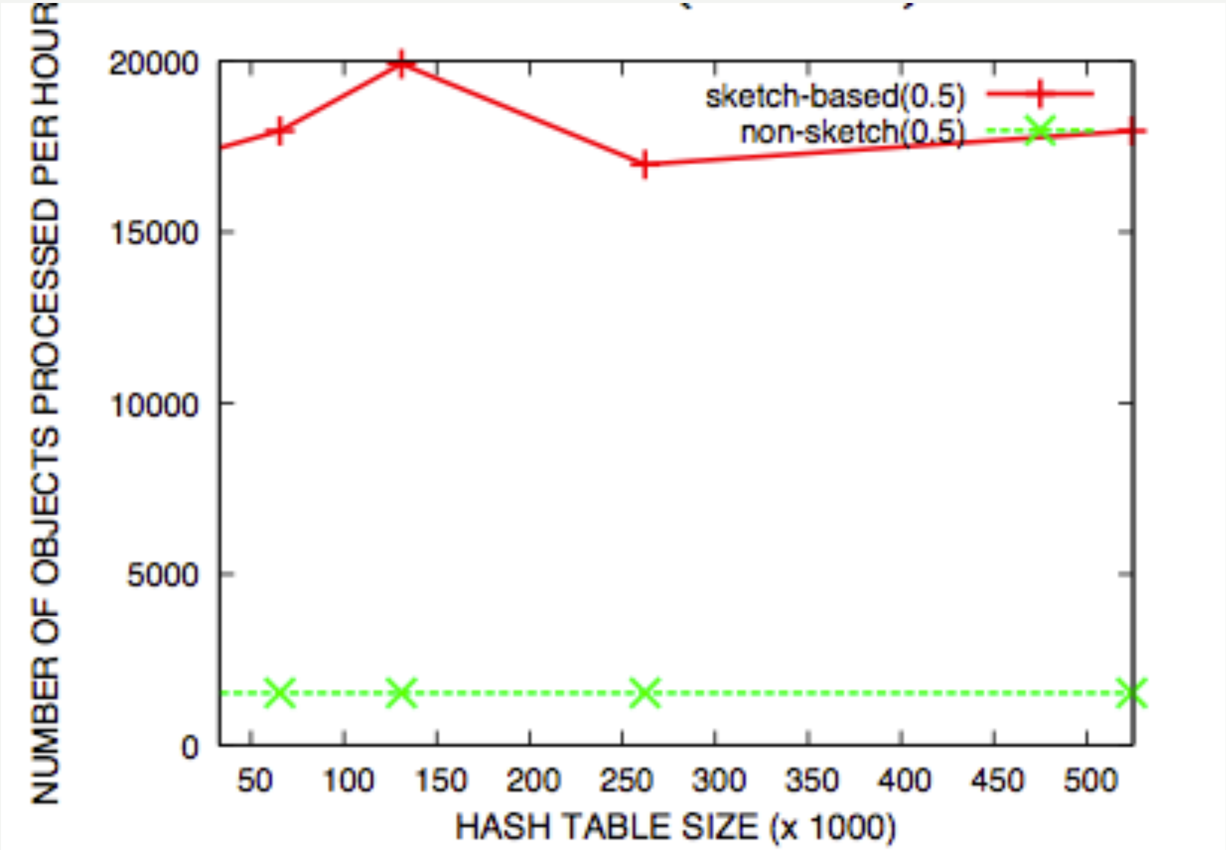


clusters
k = 750

sketch
w = 262,213
h = 2

V. Performance eval.

Efficiency of the clustering



clusters
k = 500

V. Performance eval.

Effectiveness of the event detection (unsupervised)

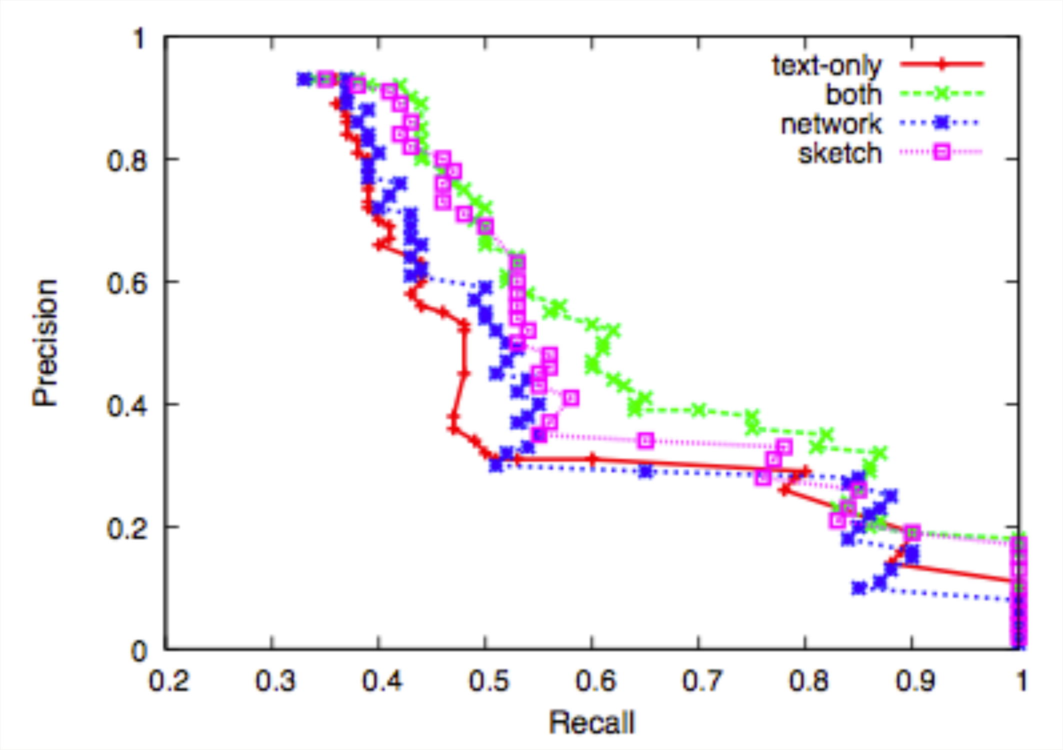
- Examined within a case study
- Detection of novel and evolution events succeeded
- Detection even with foreign language content
- Connection of related events succeeded

Effectiveness of the event detection (supervised)

- For each period of 5 minutes an event bit was set
- Continuous alarm signal fires at given threshold t
- Variation in threshold t
- Results in tradeoff between precision and recall

V. Performance eval.

Effectiveness of the event detection (supervised)



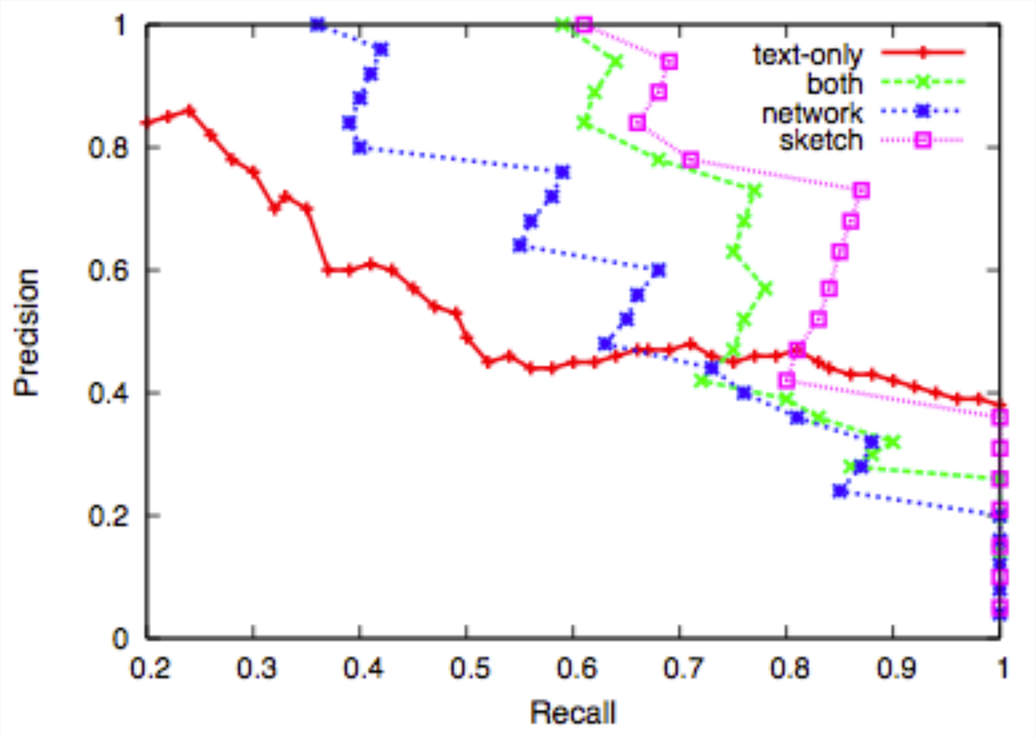
Japan Nuclear Crisis

clusters
k = 750

sketch
w = 262,213
h = 2

V. Performance eval.

Effectiveness of the event detection (supervised)



clusters
k = 750

sketch
w = 262,213
h = 2

Uganda protests

VI. Summary

++ Usage of content and structure

\+ Efficient methods (effective speed up)

\+ Effective methods

\- Novel event criteria questionable

\- Focus of paper on clustering rather than event detection