Automatic email filters generation

18.337 project

12/12/12

Ira Zhelavskaya

Presentation outline

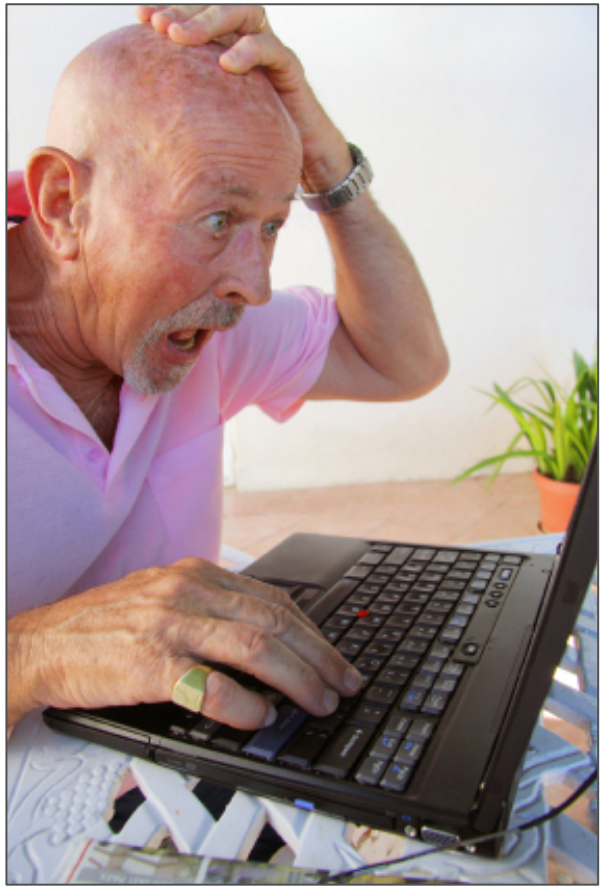
- Problem
- Idea
- Algorithm
- Opportunities for parallelization
- Results
- Future Work

Problem

Problem


Deleted Items
Drafts
Inbox (99999)
Outbox (1)
Sent Items
MMS
Outlook E-mail

Problem



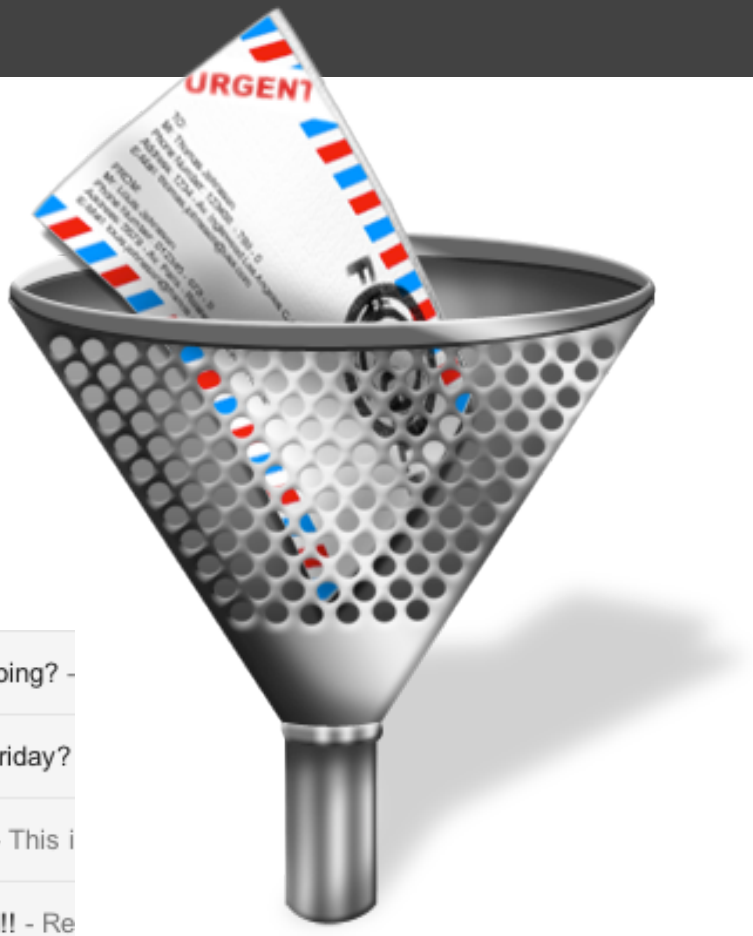

Deleted Items
Drafts
Inbox (99999)
Outbox (1)
Sent Items
MMS
Outlook E-mail

Problem

Problem

Peter, me (5)	Urgent!	How are things going? -
Simone Davids	To Do	Up for a concert Friday?
Phil Sharp		Just got my chromebook! - This i
me, Peter (2)	Hiking	Hike this weekend!! - Re

Problem



Peter, me (5)	Urgent!	How are things going? -
Simone Davids	To Do	Up for a concert Friday?
Phil Sharp		Just got my chromebook! - This i
me, Peter (2)	Hiking	Hike this weekend!! - Re

Problem

Idea

- To create a program that would automatically generate filters

Idea

- To create a program that would automatically generate filters
  - basing on the information that we can get from how a user arrange his present mail in folders.

Solutions

Solutions

- Install mail client with option of automatic email categorization

Solutions

- Install mail client with option of automatic email categorization
  - Install it

Solutions

- Install mail client with option of automatic email categorization
  - Install it
  - Not convenient to use with several devices.

Solutions

- Install mail client with option of automatic email categorization
  - Install it
  - Not convenient to use with several devices.
- Gmail Smart Folders

Solutions

- Install mail client with option of automatic email categorization
  - Install it
  - Not convenient to use with several devices.
- Gmail Smart Folders
  - Mass mailing only

Solutions

- Install mail client with option of automatic email categorization
  - Install it
  - Not convenient to use with several devices.
- Gmail Smart Folders
  - Mass mailing only
  - Not an individual filtering.

High-level idea

- To create Gmail plugin that will process user's mail and use the information about folders/tags/labels to generate filters.

Algorithm

Algorithm

- Definitions

Algorithm

- Definitions
- Algorithm description

Definitions

Definitions

- Mail (a letter) is a vector:

$$\boldsymbol{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k),$$

where $\mathbf{x}_i$ is a letter attribute(field), $i=1..k$, k – number of attributes (fields).

Definitions

- Mail (a letter) is a vector:

$$\boldsymbol{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k),$$

where $\mathbf{x}_i$ is a letter attribute(field), $i=1..k$, k – number of attributes (fields).

- Field is a vector:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in\_i}),$$

x_{ij} is a field value (unique identifier), n_i – is a number of values of the i-th attribute, j is a number of this attr.

Definitions

- Folder / tag

Suppose we have a set of M letters (inbox) $\boldsymbol{X_m}$, $m=1,..,M$. There also exist S folders $P_1, P_2, \ldots, P_S$ such that

$$\forall\ m\ \exists !s: \boldsymbol{X_m} \in P_s \Leftrightarrow P_i \cap P_j = \varnothing,\ i \neq j$$

Definitions

- Folder / tag

Suppose we have a set of M letters $\boldsymbol{X_m}$, $m=1,..,M$. There also exist S folders $P_1, P_2, \ldots, P_S$ such that

$$\forall\ m\ \ \exists !s: \boldsymbol{X_m} \in P_s \Leftrightarrow P_i \cap P_j = \varnothing,\ i \neq j$$

- Filter is a vector:

$$\boldsymbol{F} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_k),$$

where $\boldsymbol{f}_i$ is a field of a filter, $i=1..k$, k – number of attributes (fields).

Formal problem statement

- Given:

Inbox: set of M letters $\boldsymbol{X_m}$, $m=1,..,M$. It is known that letters belongs to S folders P_1, P_2, …,P_S.

Formal problem statement

- Given:

M letters $\boldsymbol{X}_m$, $m=1,..,M$. It is known that letters belongs to S folders $P_1, P_2, \ldots, P_S$.

- Find:

Such set of filters $\{\boldsymbol{F}_s\}$, $s=1,..,S$:

$$\forall \boldsymbol{X}_m \in \boldsymbol{P_s} \;\; \forall i \;\; \exists t \in 1,..,n\_i^m : \rightarrow x_{it} \in \boldsymbol{f}_{si}$$

and

$$\forall \boldsymbol{X}_m \notin \boldsymbol{P_s} \;\; \exists i \;\; \forall t \in 1,..,n\_i^m : \rightarrow x_{it} \notin \boldsymbol{f}_{si}$$

Live example

- Letter

Live example

- Letter:

From: t850@mail.com

T-850 model 101 Terminator

Live example

- Letter:

From: t850@mail.com

T-850 model 101 Terminator

To: jconnor@mail.com

John Connor

Subject: I will save you

Tag: Family

Live example

- Letter:

From: t850@mail.com $\Leftrightarrow \mathbf{x}_1 = (x_{11})$ – *value #1* of the *1*-st field (sender)

T-850 model 101 Terminator

To: jconnor@mail.com $\Leftrightarrow \mathbf{x}_2 = (x_{21})$

John Connor

Subject: I will save you $\Leftrightarrow \mathbf{x}_3 = (x_{31}, x_{32}, x_{33}, x_{34})$, $n\_3 = 4$.

Tag: Family $\Leftrightarrow \mathbf{x}_4 = (x_{41})$ – *value #1* of the *4-th field*

Live example: algorithm description

Let's consider one tag/folder – family (#*i*).

Live example: algorithm description

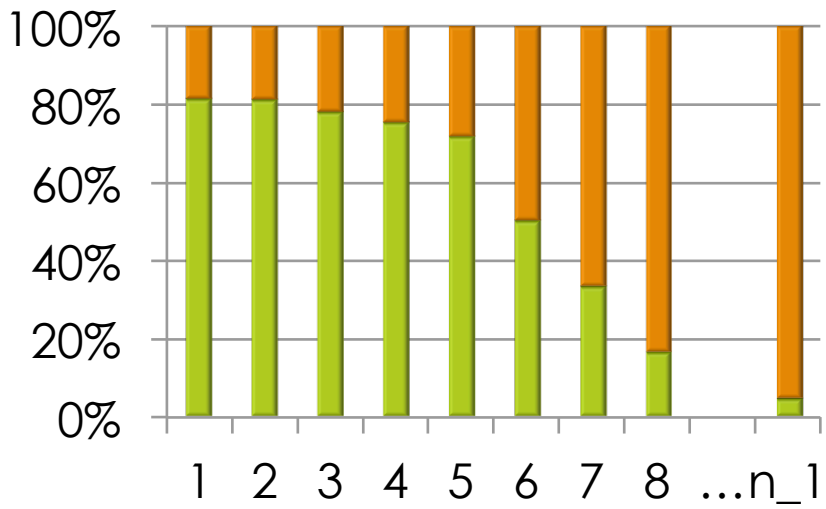
Let's consider one tag/folder (#*i*).

1. Build normalized frequency histograms for each letter's field (for all letters in the folder).

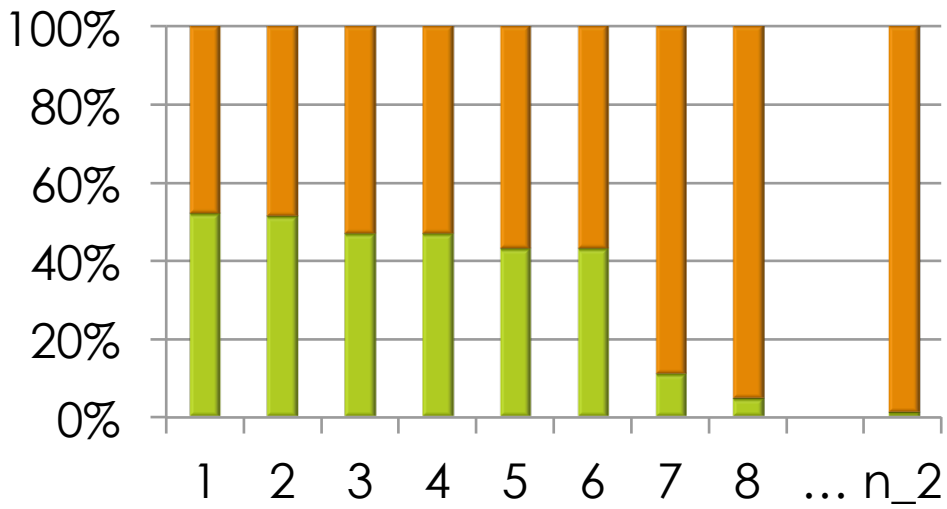
Histograms

Tag: Family

Attr.1: Sender



Attr.2: Subject



Vocabulary

1: Sarah Connor
2: Kate Brewster
3. T 850
…

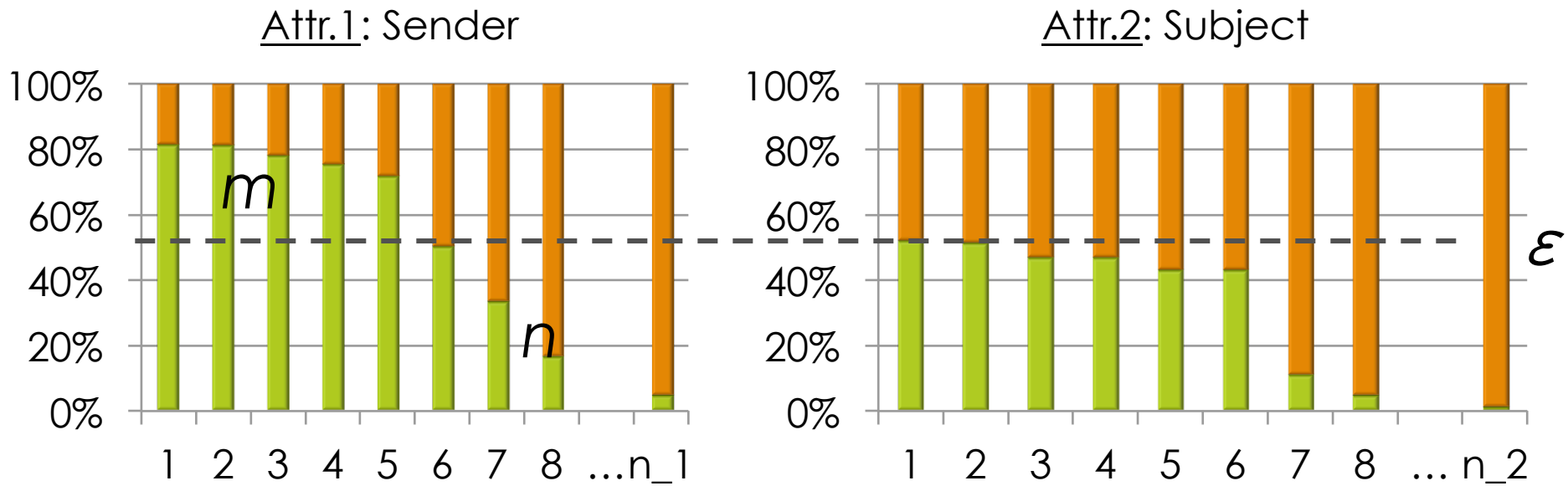
…

Live example: algorithm description

Let's consider one tag/folder (#*i*).

1. Build normalized frequency histograms for each letter's field (for all letters in the folder).
2. Analyze histograms → choose the "best" histograms

Histograms


Attr.1: Sender
100%
80%
60%
40%
20%
0%
m
n
1 2 3 4 5 6 7 8 …n_1
Attr.2: Subject
100%
80%
60%
40%
20%
0%
ε
1 2 3 4 5 6 7 8 … n_2

Live example: algorithm description

Let's consider one tag/folder (#*i*).

1. Build normalized frequency histograms for each letter's field (for all letters in the folder).

2. Analyze histograms → choose the "best" histograms (where m/n → **max**).
   1. If m/n > ε : choose this attribute value for the filter,
   2. Else, take the rest columns and build histograms for them for other remaining attributes.
   3. Go to 2.1.

Opportunities for parallelization

- Tags are processed in parallel,
- Mail preprocessing also can be parallelized,
- Inner parallelization of histograms count,
- Parallel word count (Map Reduce),
- Large inbox (= large matrix) can be stored distributed and processed in parallel.

Results

- Implementation
  - Matlab for the whole algorithm
  - Julia for the core function
- Experiments on both real and synthetic datasets
  - Data was generated from Naïve Bayes model
  - Cross-validation was used to evaluate the error rate
  - ~5% error rate on real and synthetic data
- Speedup with parallelization
  - Parallel processing of each folder (tag)
  - 160 sec vs. 0.1 sec in Julia…

Future work

- Investigate more parallelization options
- Compare with more algorithms
- Improve existing method
- Make a plugin for Gmail

Thank you! ☺

Project goal

- To write a program that will allow to create filters for a mailbox,
  - With a purpose to use it in Gmail as a toolbox for filters generation,
    - It is not a usual letters classification,
      - It is generation of filters like Gmail ones.