Predicting Web 2.0 Thread Updates
Progress Update

Shawn Tan

Table of Contents

Motivation

- Many sites with thread-based discussion features
- Users post product reviews, feedback

Obtaining such up-to-date information may be vital to companies.

	T	FB L	FB S	G +1	L	DL	C	PV	Follows
http://www.lifehacker.com	1	1	1				1	1	
http://digg.com/	1	1			1	1	1	1	
http://9gag.com/	1	1	1	1	1		1	1	
http://www.flickr.com/					1		1	1	
http://news.ycombinator.com/					1		1		
http://stackoverflow.com/					1		1	1	
http://www.youtube.com/					1	1	1	1	
http://www.reddit.com/					1	1	1		
http://www.stumbleupon.com/					1		1	1	
http://delicious.com/	1	1					1	1	1

Table: Features of popular Web 2.0 sites

T = Twitter mentions
FB L = Facebook Likes
FB S = Facebook Shares
G +1 = Google +1
L = Likes (Local)
DL = Dislikes (Local)
C = Comments
PV = Page Views
Follows = Site-local feature for keeping track of user's activities

Ideally, an incremental crawler of such user-generated content should be able to maintain fresh content.

Crawling forums: The Naive way

One way of keeping the database fresh, is to download pages at a frequent rate.
However, forum sites are too large, with too many threads, incurring high bandwidth costs.
Example: `forums.hardwarezone.com.sg`

- EDMW already has a total of 318615 threads[1].

[1] `http://forums.hardwarezone.com.sg/sitemap/f-16-p-1368.html`

Using naive method:

1. incur excessive costs when downloading un-updated pages
2. raise the possibility of the web master blocking the requester's IP address.

Crawling forums: Estimating Future posts

Attempt to estimate future posts by learning from intervals between past posts.

Extending current work

Use the content as well to attempt to make a better prediction.

- Technical forum discussions
- Flame wars (e.g. Vim vs. Emacs)

Threads in general have word signals that may hint at a different rate of updates.

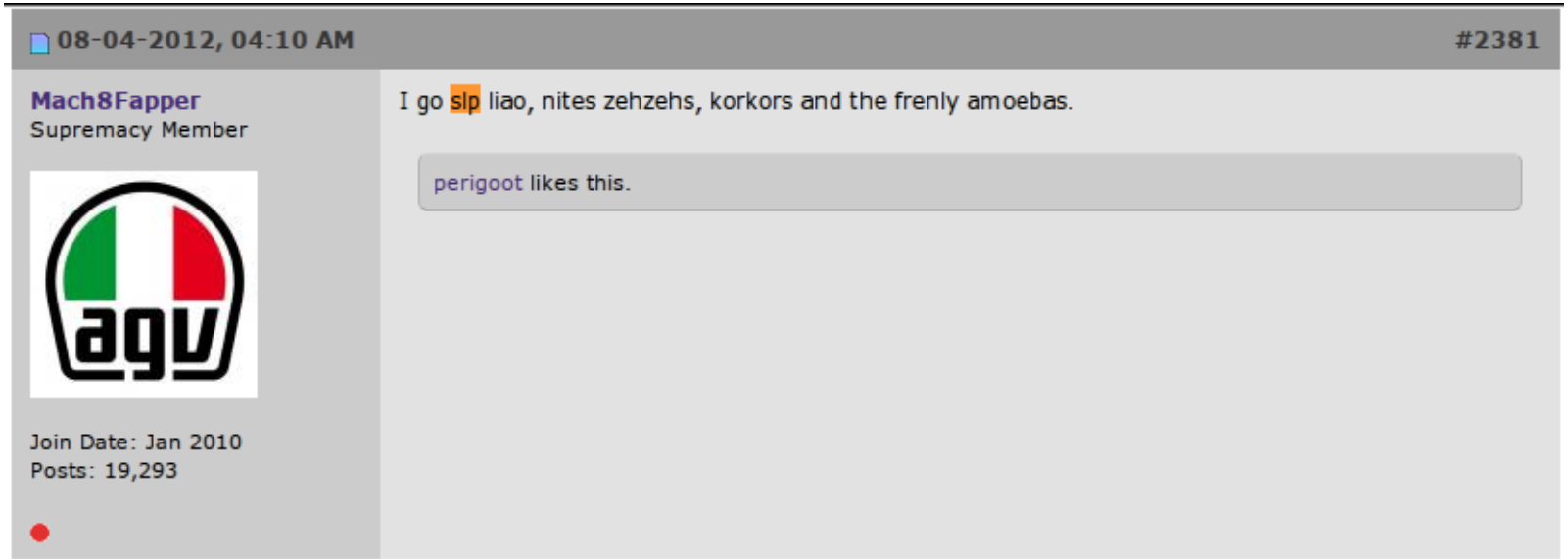


Table of Contents

Refresh policies for incremental crawlers

Many works have used time difference to estimate page updates.

1. Coffman et. al. 1997 analysed the theoretical aspects.
2. Cho and Garcia-Molina trace the change history of 720,000 web pages collected over 4 months.
    1. Showed empirically that the Poisson process model estimates the update processes well (Cho et. al. 1999)
    2. Proposed different revisiting or refresh policies (Cho et. al. 2003, Garcia-molina et. al. 2003)
3. Also used in Tan et. al. 2007.

Problems with Poisson

The Poisson distribution is memoryless, and in experimental results due to Brewington and Cybenko 2000, the behaviour of site updates are not.

Using Site-level Knowledge

Yang et. al. 2009, attempted to resolve this by

1. Using the list structure of forum sites to infer a sitemap.
2. Use a linear-regression model to predict when the next update to the thread will arrive.
3. Linear model used together with the sitemap information to prioritise the request queue in the crawler.
4. Has the ability to make use of index information to infer changes in threads. Other types of comment systems do not have such indices.

Summary

- Previous work dealt with Web 1.0 sites.
- Did not take into account the content in the posts.
- Evidence to show that using time, while makes reasonable prediction, does not fully model the behaviour of threads.

Table of Contents

Extracting Data

Collected data from `forums.hardwarezone.com.sg`, `avsforum.com` as our data set.

- User
- Timestamp
- Message body

Preliminary experiments

We picked a few threads (more than 3 days):

1. Computed time difference between posts Δt

$$\cdots \quad p_i \underbrace{\qquad}_{\Delta t_i} p_{i+1} \quad \cdots$$

2. Sort posts by time difference
3. Use median of Δt as splitting point
4. Train a Naive Bayes classifier using bag-of-words model to classify posts into 2 categories:
   - $\Delta t > 6$
   - $\Delta t \leq 6$

Results

Class	Precision	Recall	F_1
$\Delta t \leq 6$	0.657	0.816	0.728
$\Delta t > 6$	0.682	0.483	0.565
Overall	0.670	0.650	0.647

Table: Naive Bayes classification results

1. 10-fold cross validation for results
2. Low recall value for $\Delta t > 6$. May be due to overlapping terms in vocabulary resulting in $\Delta t > 6$ posts being wrongly classified.

Suggests that there is a relationship between content and update rates.

Yang et. al. 2009 Linear model

We wanted to evaluate how well the linear model performs without the sitemap information:

1. Implemented linear model from Yang et. al. 2009
    1. Used time based features (e.g. Average Δt, Day of week, Hour of day)
2. Trained against data from `http://avsforum.com`
3. Compared against model that uses the average previous 5 Δt values.

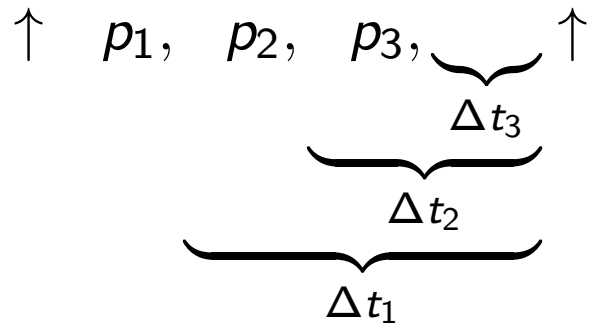
Modifications to model

1. Used baseline method (average of previous 5) if linear model returns a negative value.
2. Revisits at the same predicted time interval when there's no change to the thread.

Evaluation metric

To evaluate *timeliness* we use the metric employed in Yang et. al. 2009

$$T = \frac{1}{N} \sum_{i=1}^{N} \Delta t_i$$



Problem: A crawler that hits the site repeatedly performs well according to this metric.

1. Limited by daily bandwidth (No. of pages)
2. If bandwidth used up, downloads the next day, which contributes to overall T

Results

The linear model performed worse than the 5-average model. Values rounded up to nearest minute:

- Average Baseline Timeliness: 2490 minutes
- Average LM Timeliness: 3469 minutes
- Average paired difference (LM – Baseline): 980 minutes

Model working as part of the larger overal framework as seen in Yang et. al. 2009 may give better results than just the linear model by itself.

Table of Contents

Week 3 - Week 4

Implementation of baselines

- Yang et. al. 2009 linear regression
- Discretised values using classification
  - Try to predict classes or bins of Δt values

Week 5 - Week 9

Explore other possibilities for prediction.
Some considerations:

1. Adaptive model
2. Efficiency
3. Predicting Responses to Microblog Posts (Artzi et. al. 2012)
   - Use social data

Week 10 (Optimistic) - Week 13

Evaluation of the method if completed.