1. Social Media
Outline

1.1. What is Social Media?
1.2. Opinion Retrieval
1.3. Feed Distillation
1.4. Top-Story Identification
1.1. What is Social Media?

- Content creation is **supported by software** (no need to know HTML, CSS, JavaScript)
- Content is **user-generated** (as opposed to by big publishers) or **collaboratively-edited** (as opposed to by a single author)
- **Web 2.0** (if you like –outdated– buzzwords)
- **Examples:**
  - **Blogs** (e.g., Wordpress, Blogger, Tumblr)
  - **Social Networks** (e.g., facebook, Google+)
  - **Wikis** (e.g., Wikipedia but there are many more)
  - …
Weblogs, Blogs, the Blogosphere

- **Journal-like website**, editing supported by software, self-hosted or as a service
- Initially often run by **enthusiasts**, now also common in the **business world**, and some bloggers make their living from it
- **Reverse chronological order** (newest first)
- **Blogroll** (whose blogs does the blogger read)
- **Posts** of varying length and topics
- **Comments**
- Backed by **XML feed** (e.g., RSS or Atom) for **content syndication**
Weblogs, Blogs, the Blogosphere

- WordPress.com
  - ~ 60M blogs
  - ~ 50M posts/month
  - ~ 50M comments/month

- Tumblr.com (by Yahoo!)
  - ~ 208M blogs
  - ~ 95B posts
  - ~ 100M posts/day

- Blogger.com (by Google)
Twitter

- Micro-blogging service created in March ‘06
- Posts (tweets) limited to 140 characters
- 271M monthly active users
- 500M tweets/day = ~6K tweets/second
- 2B queries per day
- 77% of accounts are outside of the U.S.
- Hashtags (#atir2014)
- Messages (@kberberi)
- Retweets
Facebook, Google+, LinkedIn, Pinterest, …
Facebook, Google+, LinkedIn, Pinterest, …
Challenges & Opportunities

- **Content**
  - **plenty of context** (e.g., publication timestamp, relationships between users, user profiles, comments)
  - **short posts** (e.g., on Twitter), **colloquial/cryptic language**
  - **spam** (e.g., splogs, fake accounts)

- **Dynamics**
  - **up-to-date content** – real-world events covered as they happen
  - **high update rates** pose severe engineering challenges (e.g., how to maintain indexes and collection statistics)
How do People Search Blogs?

Mishne and de Rijke [8] analyzed a month-long query log from a blog search engine (blogdigger.com) and found that

- queries are mostly informational (vs. transactional or navigational)
  - contextual: in which context is a specific named entity (i.e., person, location, organization) mentioned, for instance, to find out opinions about it
  - conceptual: which blogs cover a specific high-level concept or topic (e.g., stock trading, gay rights, linguists, islam)

- contextual more common than conceptual both for ad-hoc and filtering queries
- most popular topics: technology, entertainment, and politics
- many queries (15–20%) related to current events
How do People Search Twitter?

- Teevan et al. [10] conducted a survey (54 MS employees), compared query logs from web search and Twitter, finding that queries on Twitter
  - are often related to celebrities, memes, or other users
  - are often repeated to monitor a specific topic
  - are on average shorter than web queries (1.64 vs. 3.08 words)
  - tend to return results that are shorter (19.55 vs. 33.95 words), less diverse, and more often relate to social gossip and recent events
- People also directly express information needs using Twitter: 17% of tweets in the analyzed data correspond to questions
10,000ft

- **Feeds** (e.g., blog, twitter user, facebook page)
- **Posts** (e.g., blog posts, tweets, facebook posts)

We’ll consider

- **textual content** of posts
- **publication timestamps** of posts
- **hyperlinks** contained in posts

We’ll ignore

- other links (e.g., friendship, follower/followee)
- hashtags, images, comments
Tasks

- **Post retrieval** identifies posts relevant to a specific information need (e.g., how is life in Iceland?)

- **Opinion retrieval** finds posts relevant to a specific named entity (e.g., a company or celebrity) which express an opinion about it.

- **Feed distillation** identifies feeds relevant to a topic, so that the user can subscribe to their posts (e.g., who tweets about C++?).

- **Top-story identification** leverages social media to determine the most important news stories (e.g., to display on front page).
1.2. Opinion Retrieval

- **Opinion retrieval** finds posts *relevant* to a specific *named entity* (e.g., a company or celebrity) which *express an opinion* about it.

- **Examples:** (from TREC Blog track 2006)
  - macbook pro
  - jon stewart
  - whole foods
  - mardi gras
  - cheney hunting

- **Title:**
  - whole foods

- **Description:**
  - Find opinions on the quality, expense, and value of purchases at Whole Foods stores.

- **Narrative:**
  - All opinions on the quality, expense and value of Whole Foods purchases are relevant. Comments on business and labor practices or Whole Foods as a stock investment are not relevant. Statements of produce and other merchandise carried by Whole Foods without comment are not relevant.

- **Standard retrieval models** can help with finding relevant posts; but how to determine *whether a post expresses an opinion*?
Opinion Dictionary

- What if we had a **dictionary of opinion words**? (e.g., like, good, bad, awesome, terrible, disappointing)

- Lexical resources with **word sentiment information**
  - SentiWordNet ([http://sentiwordnet.isti.cnr.it/](http://sentiwordnet.isti.cnr.it/))
  - General Inquirer ([http://www.wjh.harvard.edu/~inquirer/](http://www.wjh.harvard.edu/~inquirer/))
  - OpinionFinder ([http://mpqa.cs.pitt.edu](http://mpqa.cs.pitt.edu))
Opinion Dictionary

- He et al. [4] construct an opinion dictionary from training data
  - consider only words that are neither too frequent (e.g., and, or) nor too rare (e.g., aardvark) in the post collection D
  - let \( D_{rel} \) be a set of relevant posts (to any query in a workload) and \( D_{relopt} \subset D_{rel} \) be the subset of relevant opinionated posts
  - two options to measure opinionatedness of a word \( v \)
    - Kullback-Leibler Divergence
      \[
      op_{KLD}(v) = P[v \mid D_{relopt}] \log_2 \frac{P[v \mid D_{relopt}]}{P[v \mid D_{rel}]}
      \]
    - Bose Einstein Statistics
      \[
      op_{BO}(v) = tf(v, D_{relopt}) \log_2 \frac{1 + \lambda}{\lambda} + \log_2 (1 + \lambda) \quad \text{with} \quad \lambda = \frac{tf(v, D_{rel})}{|D_{rel}|}
      \]
Re-Ranking

- He et al. [4] measure opinionatedness of a post \( d \) as follows
  - consider the set \( Q_{\text{opt}} \) of \( k \) most opinionated words from the dictionary
  - issue \( Q_{\text{opt}} \) as a query (e.g., using Okapi BM25 as a retrieval model)
  - the retrieval status value \( \text{score}(d, Q_{\text{opt}}) \) measures how opinionated \( d \) is

- Posts are ranked in response to query \( Q \) (e.g., whole foods) according to a (linear) combination of retrieval scores

\[
\text{score}(d) = \alpha \cdot \text{score}(d, Q) + (1 - \alpha) \cdot \text{score}(d, Q_{\text{opt}})
\]

with \( 0 \leq \alpha \leq 1 \) as a tunable mixing parameter
Huang and Croft [5] expand the query with query-independent ($Q_I$) and query-dependent ($Q_D$) opinion words; posts are then ranked according to

$$score(d) = \alpha \cdot score(d, Q) + \beta \cdot score(d, Q_I) + (1 - \alpha - \beta) \cdot score(d, Q_D)$$

with $0 \leq \alpha, \beta \leq 1$ as a tunable mixing parameters and retrieval scores based on language model divergences.

Query-independent opinion words are obtained as:

- **seed words** (e.g., good, nice, excellent, poor, negative, unfortunate, …)
- **most frequent words** in opinionated corpora (e.g., movie reviews)
Sentiment Expansion

- **Examples**: (of most frequent words in different corpora)
  - Cornell movie reviews: like, even, good, too, plot
  - MPQA opinion corpus: against, minister, terrorism, even, like
  - Blog06(op): like, know, even, good, too

- **Observation**: Query-independent opinion words are very general (e.g., like, good) or specific to the corpus (e.g., minister, terrorism)
Query-dependent opinion words are obtained as words that frequently co-occur with query terms in pseudo-relevant documents (following the approach by Lavrenko and Croft [6]).

Given a query $q$, identify the set of $R$ of top-$k$ pseudo-relevant documents, and top-$n$ words having highest probability

$$P[w | R] \propto \sum_{d \in R} P[w | d] \prod_{v \in q} P[v | d, w]$$

$$P[v | d, w] = \left\{ \begin{array}{ll} \frac{tf(v,d)}{\sum_u tf(u,d)} & : w \in d \\ 0 & : \text{otherwise} \end{array} \right.$$
Examples: (of query-dependent opinion words)

- **mozart** → (like, good, too, even, death, best, great, genius)
- **allianz** → (best, premium, great, value, traditional, fidelity)
- **wikipedia** → (like, open, good, know, free, great, knowledge)
1.3. Feed Distillation

- **Feed distillation** identifies feeds (e.g., blogs, Twitter users) that are **relevant** to a **specific (typically rather broad) topic**

- **Examples**: (from TREC Blog track 2007)
  - movie review
  - firearm control
  - baseball
  - garden
  - mobile phone

- **Challenges**: How to capture whether a blog **consistently covers** the given topic? How to bridge **vocabulary gap** to posts?
Language Models

- Weerkamp et al. [11] develop two approaches to feed distillation estimating **language models** for **entire blog(ger)s** and **individual posts**, respectively.

- **Notation:**
  - A blog $b$ is a set of posts; $|b|$ is the number of posts by $b$.
  - A post $p$ is a bag of terms.
  - $tf(v, p)$ denotes the term frequency of term $v$ in post $p$.
  - $B$ denotes a virtual post concatenating all posts from all blogs.
Blogger Model (BM)

- Estimates a language model for each blog(ger) \( b \)
  \[
P[q | \theta_b] = \prod_{v \in q} P[v | \theta_b] \cdot tf(v,q)
\]

- Smooths probability estimates using the collection of blogs \( B \)
  \[
P[v | \theta_b] = (1 - \lambda_b) \cdot P[v | b] + \lambda_b \cdot P[v | B]
\]

with blog-specific smoothing parameter
  \[
  \lambda_b = \frac{\beta}{(1/|b| \cdot \sum_{p \in b} \sum_v tf(v,p)) + \beta}
\]

thus smoothing blogs with shorter posts more aggressively
Blogger Model

- **Two-step generation** of term $v$ from blog $b$

$$P[v | b] = \sum_{p \in b} P[v | p, b] P[p | b]$$

assuming **conditional independence** of terms given blog

$$P[v | b] = \sum_{p \in b} P[v | p] P[p | b]$$

2. Draw term from post 1. Draw post from blog

- **Uniform probability** of posts given blog (i.e., equal importance)

$$P[p | b] = 1/|b|$$

- **Maximum-likelihood estimate**

$$P[v | p] = \frac{tf(v, p)}{\sum_w tf(w, p)}$$
**Posting Model (PM)**

- Estimates a language model for each individual post $p$
  \[ P[v | \theta_p] = (1 - \lambda_p) \cdot P[v | p] + \lambda_p \cdot P[v | B] \]

with post-specific smoothing parameter

\[ \lambda_p = \frac{\beta}{(\sum_w tf(w, p)) + \beta} \]

thus smoothing short posts more aggressively

- Maximum-likelihood estimate
  \[ P[v | p] = \frac{tf(v, p)}{\sum_w tf(w, p)} \]
Posting Model

- **Likelihood of generating query** $q$ **from language model of post** $p$

\[
P[q | \theta_p] = \prod_{v \in q} P[v | \theta_p]^{tf(v,q)}
\]

- **Two-step generation** of query $q$ from blog $b$

\[
P[q | b] = \sum_{p \in b} P[q | \theta_p] P[p | b]
\]

- **Uniform probability** of posts given blog (i.e., equal importance)

\[
P[p | b] = 1/|b|
\]
Elsass et al. [3] proposed the highly similar **Large Document Model** (~BM) and **Small Document Model** (~PM) approaches.

Focus on bridging the **vocabulary gap** between high-level topic descriptions (e.g., *garden*) and posts (e.g., *seed, flower, crop*).

**Query expansion** with terms from **pseudo-relevant documents** retrieved from different corpora (again using the method from [6])

- **Blogs** (MAP 0.266 compared to small document model 0.315)
- **Posts** (MAP 0.282)
- **Wikipedia articles** (MAP 0.314)
- **Wikipedia passages** (MAP 0.313)

NO IMPROVEMENT!
Query Expansion

- Query expansion based on anchor phrases in Wikipedia
  - issue original query $q$ against Wikipedia articles as corpus
  - consider top-$k$ and top-$n$ ($k < n$) results returned by query
  - score every anchor phrase $a$ occurring in any top-$n$ result and pointing to a document $d$ from the top-$k$ result as

$$\text{score}(a) = \sum_{(a,d)} (k - \text{rank}(d))$$

- favoring frequent anchor phrases pointing to highly ranked articles
- expand query with top-$m$ anchor phrases (MAP 0.361)
1.4. Top-Story Identification

- **Top-story identification** (another task within the TREC Blog track) aims to identify the **most important news stories for a specific day** \( d \) **based on their coverage in the blogosphere**
  - **real-time** (online, limited statistics, time critical: small lag)
  - **retrospective**: (offline, full statistics)

- **Notation**:
  - \( d \) denotes the day of interest
  - \( B_d \) is the set of posts published at day \( d \); \( p \) denotes a post
  - \( n \) denotes a news article (consisting of headline and content)
  - \( tf(v,p) \) is the term frequency of term \( v \) in post \( p \)
Top-Story Identification


- **Intuition**: “News article important if discussed by many posts”

\[
\text{Importance}(n, d) \propto KL(\theta_n \mid \mid \theta_{B_d})
\]

(Note: This is a simplified version of the approach described in [7])

- Only articles published -1/+1 around the day of interest \(d\) are considered as candidates and ranked by the approach.
Blog Post Language Model

- Language model for **blog posts published at** d is estimated as

\[
P[v \mid \theta_{B_d}] = \frac{tf(v, B_d) + \mu \cdot \frac{tf(v, B)}{\sum_w tf(w, B)}}{(\sum_w tf(w, B_d)) + \mu}
\]

using Dirichlet smoothing with the collection of all posts B
News-Story Language Model

- **Option 1**: Estimate **directly from content** of news article
  \[
  P[v | \theta_n] = \frac{tf(v, n) + \mu \cdot \frac{tf(v, N)}{\sum_w tf(w, N)}}{(\sum_w tf(w, n)) + \mu}
  \]
  using Dirichlet smoothing with the entire news collection N

- **Option 2**: Estimate from top-k **pseudo-relevant blog posts** \(B_n\) retrieved using **headline** as query and **published within -1/+1 month** of the news article; again using Dirichlet smoothing with the collection of all posts B

- **Option 3**: **Interpolate language models** estimated from news article content and top-k pseudo-relevant blog posts
Summary

- **Opinion retrieval**
  finds posts expressing an opinion about a specific named entity

- **Feed distillation**
  identifies feeds worth following for a given high-level topic

- **Top-story identification**
  spots most important news articles based on coverage in blogs

- **Vocabulary gaps**
  are a common obstacle in IR but can often be bridged

- **Language models**
  are versatile and can be used to address many (if not most) tasks
<table>
<thead>
<tr>
<th>References</th>
</tr>
</thead>
</table>
*Time is of the Essence: Improving Recency Ranking Using Twitter Data*, 
WWW 2010 |
| [2] M. Efron: 
*Information Search and Retrieval in Microblogs*, 
*Retrieval and Feedback Models for Blog Feed Search*, 
SIGIR 2008 |
| [4] B. He, C. Macdonald, J. He, Iadh Ounis: 
*An Effective Statistical Approach for Blog Post Opinion Retrieval*, 
CIKM 2008 |
| [5] X. Huang and W. B. Croft: 
*A Unified Relevance Model for Opinion Retrieval*, 
CIKM 2009 |
| [6] V. Lavrenko and W. B. Croft: 
*Relevance-Based Language Models*, 
SIGIR 2001 |
References

Identifying top news stories based on their popularity in the blogosphere,  
Information Retrieval 17:326–350, 2014

[8] G. Mishne and M. de Rijke:  
A Study of Blog Search,  
ECIR 2006

Information Retrieval on the Blogosphere,  

#TwitterSearch: A Comparison of Microblog Search and Web Search,  
WSDM 2011

Blog feed search with a post index,  
Information Retrieval 14:515–545, 2011