Detecting Deceptive Opinion Spam using Linguistics, Behavioral and Statistical Modeling

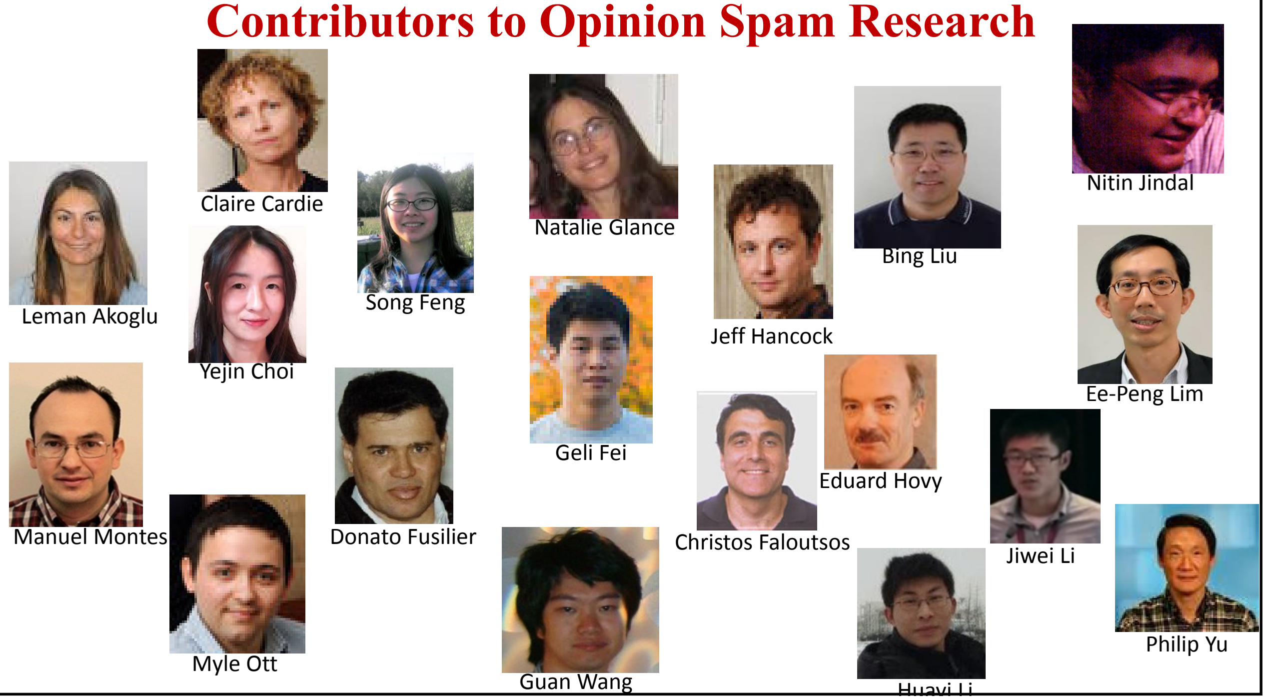
Arjun Mukherjee†
Department of Computer Science
University of Houston

† Contains contents (results/ideas/figures) of published papers by several contributors. Referenced in place.

Public opinion in this country is everything.

—Abraham Lincoln

Contributors to Opinion Spam Research



The Opinion Curse – Opinion Spam!

- Opinions $\Rightarrow$ Virtual Currency

- Opinion Spam: Illegitimate activities (e.g., writing fake reviews/ratings) to deliberatively mislead consumers
- In E-commerce, filtering opinion spam is vital

Which review is fake?

I want to make this review in order to comment on the excellent service that my mother and I received on the Serenade of the Seas, a cruise line for Royal Caribbean. There was a lot of things to do in the morning and afternoon portion for the 7 days that we were on the ship. We went to 6 different islands and saw some amazing sites! It was definitely worth the effort of planning beforehand. The dinner service was 5 star for sure. I recommend the Serenade to anyone who is looking for excellent service, excellent food, and a week full of amazing day-activities!

VS

Guacamole burger was quite tall; clam chowder was tasty. The appetizers weren't very good at all. And the service kind of lagged. A cross between Las Vegas and Disney world, but on the cheesy side. This Cafe is a place where you eat inside a plastic rain forest. The walls are lined with fake trees, plants, and wildlife, including animatronic animals. I could see it being fun for a child's birthday party (there were several that occurred during our meal), but not a place to go if you're looking for a good meal.

Which review is fake?

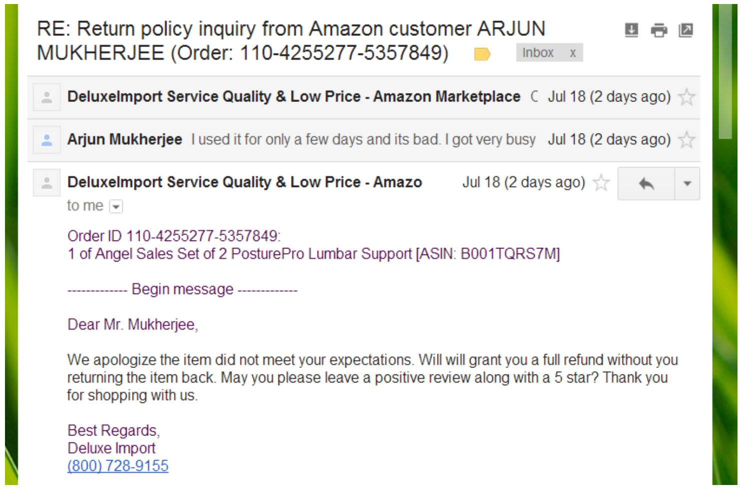
I want to make this review in order to comment on the excellent service that my mother and I received on the Serenade of the Seas, a cruise line for Royal Caribbean. There was a lot of things to do in the morning and afternoon portion for the 7 days that we were on the ship. We went to 6 different islands and saw some amazing sites! It was definitely worth the effort of planning beforehand. The dinner service was 5 star for sure. I recommend the Serenade to anyone who is looking for excellent service, excellent food, and a week full of amazing day-activities!

VS

Guacamole burger was quite tall; clam chowder was tasty. The appetizers weren't very good at all. And the service kind of lagged. A cross between Las Vegas and Disney world, but on the cheesy side. This Cafe is a place where you eat inside a plastic rain forest. The walls are lined with fake trees, plants, and wildlife, including animatronic animals. I could see it being fun for a child's birthday party (there were several that occurred during our meal), but not a place to go if you're looking for a good meal.

Opinion Spam Solicitations (Real Cases)

- I wanted to return an item I purchased from Amazon because it didn't work.
- Guess what did the seller say?


RE: Return policy inquiry from Amazon customer ARJUN MUKHERJEE (Order: 110-4255277-5357849) Inbox x

DeluxeImport Service Quality & Low Price - Amazon Marketplace Jul 18 (2 days ago)

Arjun Mukherjee I used it for only a few days and its bad. I got very busy Jul 18 (2 days ago)

DeluxeImport Service Quality & Low Price - Amazo Jul 18 (2 days ago)
to me

Order ID 110-4255277-5357849:
1 of Angel Sales Set of 2 PosturePro Lumbar Support [ASIN: B001TQRS7M]

------------- Begin message -------------

Dear Mr. Mukherjee,

We apologize the item did not meet your expectations. Will will grant you a full refund without you returning the item back. May you please leave a positive review along with a 5 star? Thank you for shopping with us.

Best Regards,
Deluxe Import
(800) 728-9155

Opinion Spam Solicitations (Real Cases)

- The case of Belkin International Inc.
- Top networking and peripherals manufacturer | Sales ~ $500 million in 2008
- Posted an ad for writing fake reviews on amazon.com (65 cents per review)


Timer: 00:00:00 of 60 minutes
Want to work on this HIT? Accept HIT
Want to see other HITs? Skip HIT

Write Product Reviews 25-50 Words
Requester: Mike Bayard
Qualifications Required: HIT approval rate (%) is not less than 95

Write a Positive 5/5 Review for Product on Website

Positive review writing.

- Use your best possible grammar and write in US English only
- Always give a 100% rating (as high as possible)
- Keep your entry between 25 and 50 words
- Write as if you own the product and are using it
- Tell a story of why you bought it and how you are using it
- Thank the website for making you such a great deal
- Mark any other negative reviews as "not helpful" once you post yours

Instructions:

The link below leads to a product on a website. Read-through the product's features and write a positive review for it using the guidelines above to the best of your ability. I have also provided the part number for this product and you can click on the links below to see it on several alternative websites. In order to post some reviews you will need to create an account on the site. You can use your own email address or open a new free webmail account (gmail, yahoo...) and use it to post with.

How Much Fake is Out There?

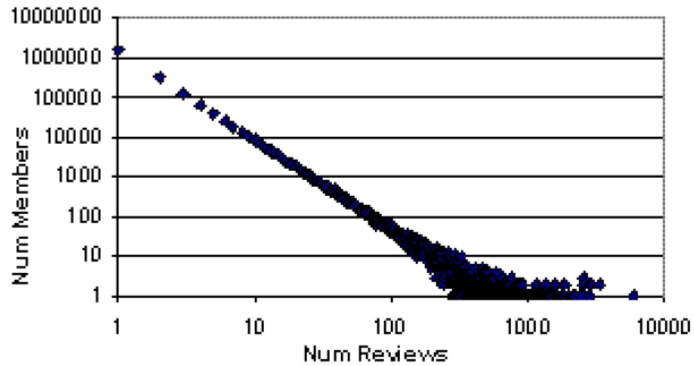
- ❑ Various estimates from different deception prevalence studies
- ❑ 2-6% in Orbitz, Priceline, Expedia, Tripadvisor, etc. [Ott et al., WWW 2012]
- ❑ 14-20% in Yelp [Mukherjee et al., ICWSM 2013; Wang et al., J. Eco. Policy 2010]

Distribution Analyses in Amazon

- ❑ Pioneering Work by [Jindal and Liu, WWW, 2007; WSDM 2008]
- ❑ Large scale analyses of opinion spam in Amazon.com
- ❑ 6M reviews, 1.2M products, 2.1M reviewers

Distribution Analyses in Amazon

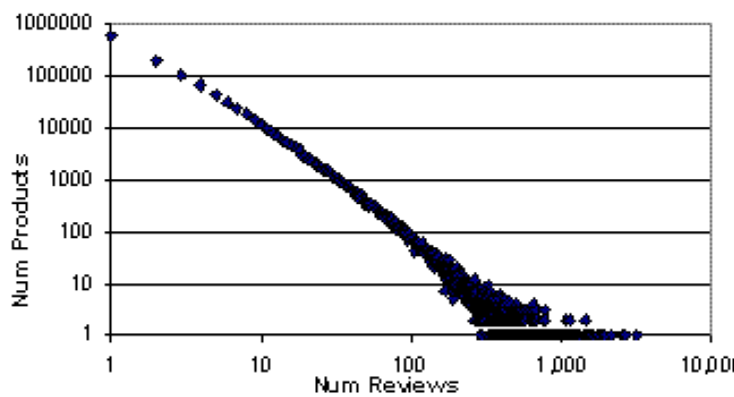
- ❑ Log-Log plot - Members vs. Reviews
- ❑ Reasonable less number of active highly active members
- ❑ Less # of extremes – reviewers with very high (potentially spam) or very low review rates



† Contains content originally appearing in [Jindal and Liu, WSDM, 2008]

Distribution Analyses in Amazon

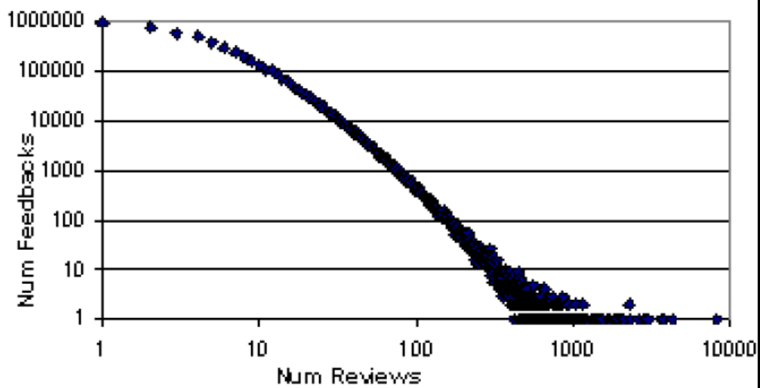
- ❑ Log-Log plot - Products vs. Reviews
- ❑ Reasonable less number of active highly active products
- ❑ Lot of products with very few reviews (potential spam targets)



† Contains content originally appearing in [Jindal and Liu, WSDM, 2008]

Distribution Analyses in Amazon

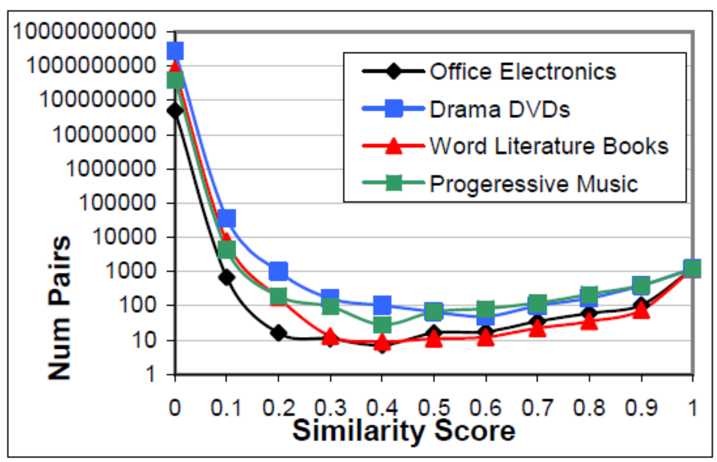
- Log-Log plot - Feedbacks vs. Reviews
- Reasonable less number of reviews with decent feedbacks
- Lot of reviews with very large helpfulness votes (potentially spam votes)



† Contains content originally appearing in [Jindal and Liu, WSDM, 2008]

Distribution Analyses in Amazon

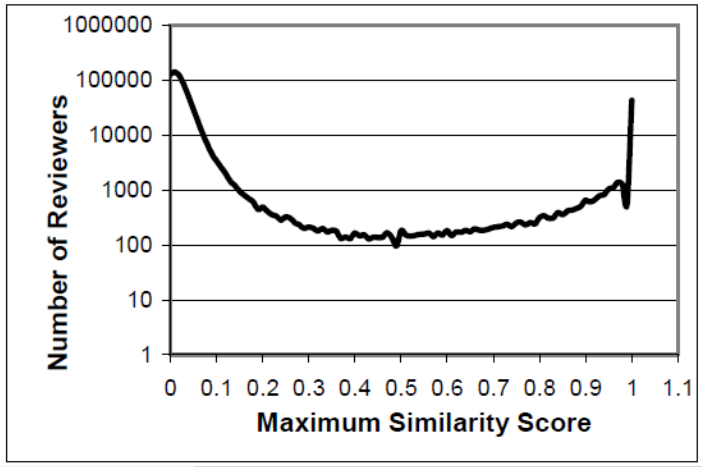
- Near duplicity analyses
- Two reviews which have similar contents are called (near) duplicates
- Spam is out there in almost every domain!



† Contains content originally appearing in [Jindal and Liu, WSDM, 2008]

Distribution Analyses in Amazon

- Near duplicity analyses – the distribution of reviewers
- Two reviews which have similar contents are called (near) duplicates
- Several spam reviewers who blatantly copy-paste reviews!



† Contains content originally appearing in [Jindal and Liu, WSDM, 2008]

Opinion Spam Types

- Type 1 (Untruthful opinions, fake reviews)
- Type 2 (Reviews on Brands Only)
  - *"I don't trust HP and never bought anything from them"*
- Type 3 (Advertisements)
  - *"Detailed product specs: 802.11g, IMR compliant, …"*
  - *"…buy this product at: compuplus.com"*

Opinion Spam Detection per Types

- Type 2, 3 → Supervised Learning
- Type 1 (Hard as difficult to get ground truths)
  - Approximation: *"Use duplicates as positive samples"*

Opinion Spam Feature Types

- Review centric features (content)
  - Features about reviews
- Reviewer centric features
  - Features about the reviewers
- Product centric features
  - Features about products reviewed.

Opinion Spam Review Feature

- Number of feedbacks (F1)
- Number (F2) and Percent (F3) of helpful feedbacks
- Length of the review title (F4)
- Length of review body (F5)
- …

Opinion Spam Reviewer Features

- Ratio of the first reviews (F22) of the products to the total number of reviews that he/she wrote
- Ratio of the number of cases in which he/she was the only reviewer (F23)
- Average rating given by reviewer (F24)
- Standard deviation in rating (F25)
- ...

Opinion Spam Product Features

- Price (F33) of the product
- Sales rank (F34) of the product
- Average rating (F35) of the product
- Standard deviation in ratings (F36) of the reviews on the product
- ...

Opinion Spam Detection Performance

- AUC of 10-fold Cross Validation
- Text features alone not sufficient
- Feedbacks unhelpful (as feedback itself subject to abuse!)

Spam Type	Num reviews	AUC	AUC – text features only	AUC – w/o feedbacks
Types 2 & 3	470	98.7%	90%	98%
Type 2 only	221	98.5%	88%	98%
Type 3 only	249	99.0%	92%	98%

Duplicate Opinion Spam Types

- ❑ Same userid, same product
- ❑ Different userid, same product
- ❑ Same userid, different products
- ❑ Different userid, different products

The last three types are very likely to be fake!

Predictive Power of Duplicates

- ❑ Representative of all kinds of spam
- ❑ Only 3% duplicates accidental
- ❑ Duplicates as positive examples, rest of the reviews as negative examples

Features used	AUC
All features	78%
Only review features	75%
Only reviewer features	72.5%
Without feedback features	77%
Only text features	63%

Near duplicates is a sheer sign of spamming – spammers usually want to recycle their fake reviews anyways!

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
  - **P.1: Deception Detection via Linguistic Classifiers**
  - **P.2: Stylometric Methods**
  - **P.3: Generic Deceptive Signals**
- ❑ Approach#2: Behavioral Modeling
- ❑ …

Linguistic Classifiers of Deception

- ❑ Deception detection via Linguistic Signals [Ott et al., ACL 2011]
- ❑ Labeling fake reviews infeasible
  - Duplicate detection [Jindal and Liu, 2008] → naïve
- ❑ Generate fake reviews using Amazon Mechanical Turk (AMT)
  - 20 hotels
  - – 20 reviews / hotel
  - – Offer $1 / review
  - – 400 reviews

Linguistic Classifiers of Deception

- Deception detection via Linguistic Signals [Ott et al., ACL 2011]
- Labeling fake reviews infeasible
  - Duplicate detection [Jindal and Liu, 2008] → naïve
- Generate fake reviews using Amazon Mechanical Turk (AMT)
  - 20 hotels
  - – 20 reviews / hotel
  - – Offer $1 / review
  - – 400 reviews

Negative samples (truthful reviews) obtained from Tripadvisor.com for those 20 hotels (length normalized)

AMT crafted fake reviews serve as Postive samples (fake reviews)

Human Performance on Deception Detection

- Test set: 80 Truthful and 80 Deceptive reviews (balanced data)
- Judges: 3 undergraduates (with 2 meta judges)
- Accuracies ranging from 53 – 61 %

Human Performance on Deception Detection

- Test set: 80 Truthful and 80 Deceptive reviews (balanced data)
- Judges: 3 undergraduates (with 2 meta judges)
- Accuracies ranging from 53 – 61 %

Deception detection is non-trivial by mere reading of reviews

Linguistic Classifier Performance Analysis

- Classifier: Linear SVM
- 3 Feature Families:
  - Genre – 48 POS tags
  - Psycholinguistics, LIWC [Pennebaker et al., 2007] – 4500 keywords across 80 linguistic dimensions
  - N-grams

Linguistic Classifier Performance

- ❑ Classifier: Linear SVM
- ❑ Accuracy results on balanced data (400+/400-) via 5-fold CV:

Feature Set	Accuracy
Genre (POS)	73.0
LIWC	76.8
Unigrams	88.4
Bigrams	89.6
Bigrams + LIWC	89.8

Outline

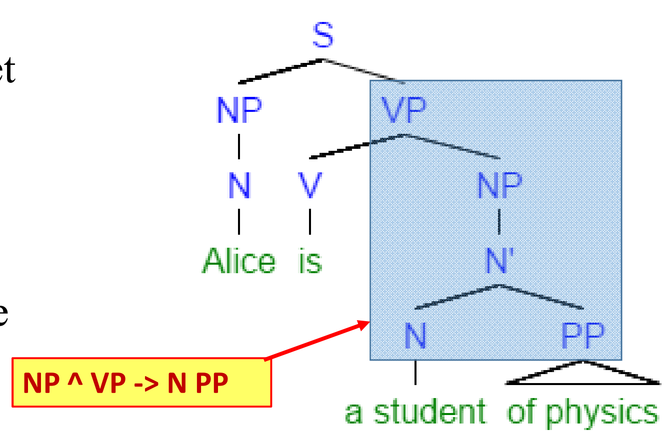
- ❑ Approach#1: Leveraging Linguistic Signals
  - **P.1: Deception Detection via Linguistic Classifiers**
  - **P.2: Stylometric Methods**
  - **P.3: Generic Deceptive Signals**
- ❑ Approach#2: Behavioral Modeling
- ❑ …

Deception Detection via Stylometry

- ❑ Syntactic Stylometry for Deception Detection [Feng et al., ACL 2012]
- ❑ Model lexicalized and unlexicalized syntactic features using sentence parse trees

Deception Detection via Stylometry

- ❑ Syntactic Stylometry for Deception Detection [Feng et al., ACL 2012]
- ❑ Model lexicalized and unlexicalized syntactic features using sentence parse trees
- ❑ Generate deep syntactic features (i.e., by rewriting production rules)



Performance Evaluation of Deep Syntax

- Classifier: LIBLINEAR
- 5-fold CV with 80-20 train-test splits
- Feature value assignment: TF-IDF
- 3 Feature Families:
  - Lexical (uni/bigrams)
  - Shallow syntax: POS tags
  - Deep syntax: rules from parse trees

Performance Evaluation of Deep Syntax

- Results due to [Feng et al., ACL 2012]
- Datasets:
  - Tripadvisor [Ott et al., ACL 2011]
  - Essay [Mihalcea and Strapparava, ACL 2009]
  - Yelp
- 3-8% improvements in Accuracy

Feature Set	TripAdvisor	Essay	Yelp
Words	88.4	77.0	59.9
Shallow Syntax	87.4	80.0	62.0
Deep Syntax	90.4	78.0	63.5
Deep Syntax + Words	91.2	85.0	64.3

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
  - **P.1: Deception Detection via Linguistic Classifiers**
  - **P.2: Stylometric Methods**
  - **P.3: Generic Deceptive Signals**
- ❑ Approach#2: Behavioral Modeling
- ❑ …

Generic Deception Signal Discovery

- ❑ General Rule for Deception Detection [Li et al., ACL 2014]
- ❑ Domains
  - Hotel
  - Restaurant
  - Doctor
- ❑ 3 Review Types
  - Turker Generated
  - Consumer Generated (Web)
  - Expert Generated

Generic Deception Signal Discovery

- General Rule for Deception Detection Result due to [Li et al., ACL 2014]
- Domains
  - Hotel
  - Restaurant
  - Doctor
- How well do deception classifiers transfer knowledge?
  - Would text classifiers trained on hotel domain work well on Doctor domain?

NYC-Hotel	0.76
Chicago Restaurant	0.77
Doctor	0.61

F1 scores of SVMs trained on [Ott et al., 2011]

Additive Models

- Tailored Additive Generative Models (e.g., extending SAGE [Eisenstein et al., EMNLP 2011])
- Capture multiple generative facets
  - Deceptive vs. truthful
  - Pos vs. Neg
  - Experienced vs. Non- Experienced
- SAGE tend to improve performances over SVMs across POS, LIWC, ngram features

Generic Deception Signals

- Main results of [Li et al., ACL 2014] inferred from estimated feature weights

- (1) Domain specific details can be predictive of deception
  - Spatial details in Hotel/Res domain reviews

- (2) Both actual customers and experts tend to include spatial details → lack of spatial details may not be a generic cue for deception

Generic Deception Signals

- Main results of [Li et al., ACL 2014] inferred from estimated feature weights

- (3) Turkers and Experts (e.g., Hotel/Res employees) tend to have an "exaggerated" use of sentiment vocabulary

- (4) Decreased use of 1st person pronouns in deceptive text – "psychological detachment" [Newman et al., 2003] – similar findings as [Mukherjee et al., ICWSM 2013]

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
- ❑ Approach#2: Behavioral Modeling
  - **P.1: Exploiting Anomalous Rating and Reviewing Behaviors**
  - **P.2: Modeling Group Spam/Collusion**
  - **P.3: Graph Based Methods**
  - **P.4: Distributional Methods**
- ❑ Approach#3: Statistical Modeling
- ❑ …

Exploiting Anomalous Reviewing Behaviors

- ❑ Linguistic models are good, but…
  - Lack in capturing fraud behaviors
  - Prone to errors/noise in discovering deception signals
  - Relatively easy to evade

- ❑ Behavior models to the rescue

- ❑ Opinions spamming invariably involves anomalous behaviors:
  - Giving unfair (high/low) ratings to entities
  - Rating burstiness (reviewing too often)
  - Colluding to spam (forming groups)
  - …

Modeling Spamming Behaviors

- Target and Deviation Based Spamming [Lim et al., CIKM 2010]

- **Observation:** Spammers direct their efforts to a set of target products/product groups and inflict spam via rating via multiple ratings

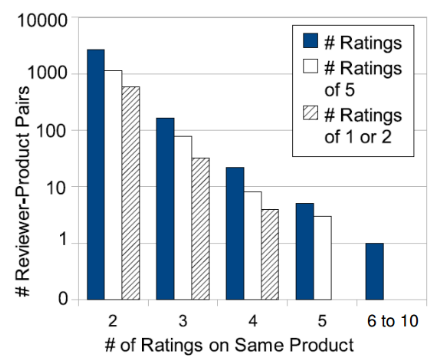
Target and Deviation Based Features

- Result on multiple ratings due to [Lim et al., CIKM 2010]

- Define $E_{ij} =$
  $set\ of\ ratings\ from\ user\ i\ to\ entity\ j$

- Figure shows the number of reviewer-product pairs (in log10 scale) having $x = |E_{ij}|\ (x > 1)$ reviews (along with distribution of all 5 ratings and pairs with all 1 or 2 ratings)

- Conclusion: several reviewers who contribute multiple reviews to the same products ⇒Target based rating spam exists



† Contains content originally appearing in [Lim et al., CIKM 2010]

Target and Deviation Based Features

- **Target-base Spamming on Products**
- Rating Spamming

$$c_{p,e}(u_i) = \frac{s_i}{Max_{u'_i \in \mathbf{U}} s_{i'}}$$

- Review Text Spamming

$$c_{p,v}(u_i) = \frac{s'_i}{Max_{u'_i \in \mathbf{U}} s'_{i'}}$$

s_i: unnormalized spam scores

- Combined Spam Score

$$c_p(u_i) = \frac{1}{2}\left(c_{p,e}(u_i) + c_{p,v}(u_i)\right)$$

$$s_i = \sum_{e_{ij} \in \mathbf{E}_{ij}, |\mathbf{E}_{ij}|>1} |\mathbf{E}_{ij}| \cdot sim(\mathbf{E}_{ij})$$

$$sim(\mathbf{E}_{ij}) = 1 - \underset{e_k, e_{k'} \in \mathbf{E}_{ij}, k<k'}{Avg} |e_k - e_{k'}|$$

$$s'_i = \sum_{v_{ij} \in \mathbf{V}_{ij}, |\mathbf{V}ij|>1} |\mathbf{V}_{ij}| \cdot sim(\mathbf{V}_{ij})$$

$$sim(\mathbf{V}_{ij}) = \underset{v_k, v_{k'} \in \mathbf{V}_{ij}, k<k'}{Avg} sim(v_k, v_{k'})$$

$$sim(v_k, v_{k'}) = cosine(v_k, v_{k'})$$

Target and Deviation Based Features

- **Deviation based Spamming**
- Single product group multiple high ratings

$$\mathrm{E}^{\mathcal{H}}_{ik}(w) = \{e_{ij} \in \mathrm{E}_{i*} \mid o_j \in b_k \wedge t(e_{ij}) \in w \wedge e_{ij} \in H\,RatingSet\}$$

$$\mathbf{C}^{\mathcal{H}}_i = \cup_{k,w}\{\mathbf{E}^{\mathcal{H}}_{ik}(w) \mid \ |\mathbf{E}^{\mathcal{H}}_{ik}(w)| \geq minsize^{\mathcal{H}}\}$$

$$c_{g,\mathcal{H}}(u_i) = \frac{\mathbf{C}^{\mathcal{H}}_i}{Max_{u'_i \in \mathbf{U}} \mathbf{C}^{\mathcal{H}}_{i'}}$$

- Single product group multiple low ratings

$$\mathrm{E}^{\mathcal{L}}_{ik}(w) = \{e_{ij} \in \mathrm{E}_{i*} \mid o_j \in b_k \wedge t(e_{ij}) \in w \wedge e_{ij} \in L\,RatingSet\}$$

$$\mathbf{C}^{\mathcal{L}}_i = \cup_{k,w}\{\mathbf{E}^{\mathcal{L}}_{ik}(w) \mid \ |\mathbf{E}^{\mathcal{L}}_{ik}(w)| \geq minsize^{\mathcal{L}}\}$$

$$c_{g,\mathcal{L}}(u_i) = \frac{\mathbf{C}^{\mathcal{L}}_i}{Max_{u'_i \in \mathbf{U}} \mathbf{C}^{\mathcal{L}}_{i'}}$$

- Combined Spam Score

$$c_g(u_i) = \frac{1}{2}\left(c_{g,\mathcal{H}}(u_i) + c_{g,\mathcal{L}}(u_i)\right)$$

Target and Deviation Based Features

- **Target-base Spamming on Products Groups**
- General Deviation

$$d_{ij} = e_{ij} - \operatorname*{Avg}_{e \in \mathbf{E}_{*j}} e \qquad c_d(u_i) = \operatorname*{Avg}_{e_{ij} \in \mathbf{E}_{i*}} |d_{ij}|$$

- Early Deviation

$$d_{ij} = e_{ij} - \operatorname*{Avg}_{e \in \mathbf{E}_{*j}} e$$

$$c_e(u_i) = \frac{\sum_{e_{ij} \in \mathbf{E}_{i*}} (|d_{ij}| \times w_{ij})}{\sum_{e_{ij} \in \mathbf{E}_{i*}} w_{ij}}$$

$$w_{ij} = \frac{1}{(r_{ij})^{\alpha}}$$

r_{ij}: rank order/position of rating j by user i

Deception Ranking

- Each method works by ranking the users by decreasing behavior score order. The highly ranked users are more likely to be spammers. Methods are:
- (a) single product multiple reviews behavior (**TP**); (b) single product group multiple reviews behavior (**TG**) only; (c) general deviation (**GD**) behavior; and (d) early deviation (**ED**) behavior with α=1.5.
- combined method (ALL) that considers all the behaviors using a combined score.
- Baseline: ranks the reviewers by their average unhelpfulness.

True Spamicity Ranking (via human experts)

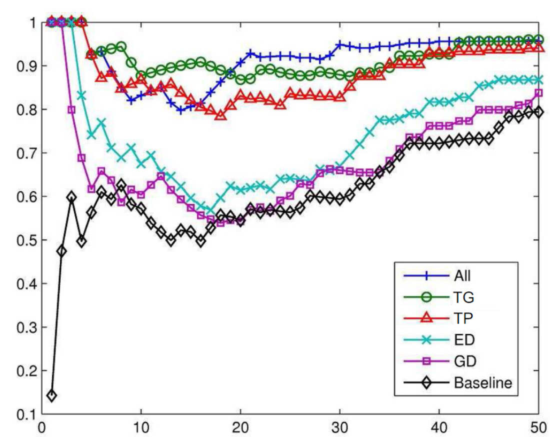
- Obtaining gold-standard spamicity ranking for each reviewer is challenging due to scale.
- **Selective evaluation:** Select 10 top/bottom ranked reviewers for each spammer detection method. Merge all the selected spammers into a pool. Upon sorting, 25 top ranked reviewers and 25 bottom ranked reviewers are then selected for user evaluation.
- The above ranking feeds the signal for DCG (of ideal ranking)

Spamicity Ranking Evaluation

- Performance of Spammer Ranking
- Result due to [Lim et al., CIKM 2010]
- NDCG: Measure Closeness to idea human expert ranking
- TP, TG, All methods tend to work well

⇒ Multi rating behaviors are discriminative

	Baseline	TP	TG	GD	ED	ALL
# spammers in top 10	7	10	10	6	6	10
# non-spammers in bottom 10	7	10	9	6	7	10



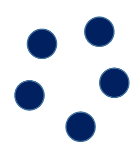
† Contains content originally appearing in [Lim et al., CIKM 2010]

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
- ❑ Approach#2: Behavioral Modeling
  - **P.1: Exploiting Anomalous Rating and Reviewing Behaviors**
  - **P.2: Modeling Group Spam/Collusion**
  - **P.3: Graph Based Methods**
  - **P.4: Distributional Methods**
- ❑ Approach#3: Statistical Modeling
- ❑ …

Modeling Collusion

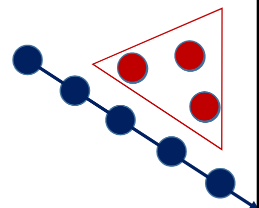
- ❑ Spammers writing fake reviews in collusion
- ❑ Prior work:
  - Deception analysis [Ott et al., ACL 2011]
  - Individual opinion spammers [Jindal & Liu, WSDM 2008; Lim et al., CIKM 2010]



- ❑ Group has more manpower→ Controls overall sentiment
- ❑ **Group context on big data can help spot anomalies**

Modeling Collusion

- Spammers writing fake reviews in collusion
- Prior work:
  - Deception analysis [Ott et al., ACL 2011]
  - Individual opinion spammers [Jindal & Liu, WSDM 2008; Lim et al., CIKM 2010]



- Group has more manpower→ Controls overall sentiment
- **Discover patterns → Spot anomalies**

Group Spam Indicators

1. Group Time Window (GTW):

$$GTW(g) = \max_{p \in P_g}(GTW_p(g,p))$$

$$GTW_p(g,p) = \begin{cases} 0; L(g,p) - F(g,p) > \tau \\ 1 - \frac{L(g,p) - F(g,p)}{\tau}; otherwise \end{cases}$$

2. Group Deviation (GD):

$$\max_{p \in P_g}(D(g,p))$$

$$D(g,p) = \frac{|r_{p,g} - \bar{r}_{p,g}|}{4}$$

3. Group Content Similarity (GCS):

$$GCS(g) = \max_{p \in P_g}(CS_G(g,p))$$

$$CS_G(g,p) = avg(cosine(c(m_i,p), c(m_j,p)))$$

4. Group Member Content similarity (GMCS):

$$GMCS(g) = avg(CS_M(g,m))$$

$$CS_M(g,m) = avg(cosine(c(m,p_i), c(m,p_j)))$$

Group Spam Indicators

5. Group Early Time Frame (GETF):
$$GETF(g) = \max_{p \in P_g}(GTF(g,p))$$
$$GTF(g,p) = \begin{cases} 0; L(g,p) - A(p) > \beta \\ 1 - \frac{L(g,p) - A(p)}{\beta}; otherwise \end{cases}$$

6. Group Size Ratio (GSR):
$$GSR(g) = avg(GSR_P(g,p))$$
$$GSR_P(g,p) = \frac{|g|}{|M_p|}$$

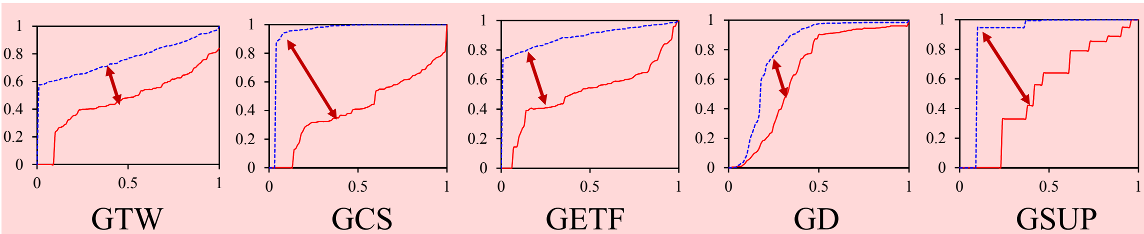
7. Group Size (GS): $GS(g) = \frac{|g|}{\max(|g_i|)}$

8. Group Support Count (GSUP): $GSUP(g) = \frac{|P_g|}{\max(|P_{g_i}|)}$

These eight group behaviors can be seen as group spamming features for learning. From here on, we refer the 8 group behaviors as $f_1 \ldots f_8$ when used in the context of features.

Cumulative Behavioral Distribution

$f \to 0+$: Normal ; $f \to 1-$: Anomalous behavior



Cumulative % of spam (solid) and non-spam (dashed) groups vs. feature value

Gaps → Discriminative strength

Modeling Group Spam – Need for Relations

- Standard feature based learning [Jindal and Liu, WSDM 2008; Ott et al., ACL 2011] falls short
- Groups share members. i.e., apart from group features, the group spamicity is also affected by other groups sharing its members, the spamicity of the shared members, etc.
- Group features (f1…f8) only summarize (e.g., by max/avg) group behaviors but individual member level spam contributions not considered.
- No notion of extent to which a product is spammed

Group Spam – Product Relation

- The group spam-product relations can be expressed as:
  - $s(p_i) = \sum_{j=1}^{|G|} w_1(p_i, g_j) s(g_j)\,; V_P = W_{PG} V_G \quad (1)$
- (1) computes the extent p_i is spammed by various groups. It sums the spam contribution by each group, $w_1(p_i, g_j)$, and weights it by the spamicity of that group, $s(g_j)$
  - $s(g_j) = \sum_{i=1}^{|P|} w_1(p_i, g_j) s(p_i)\,; V_G = W_{PG}^T V_P \quad (2)$
- (2) updates the group's spamicity by summing its spam contribution on all products weighted by the extent those products were spammed.

Member Spam – Product Relation

- Like w_1, we employ $w_2 \in [0, 1]$ to model spam contribution by a member m_k towards product p_i.
- Like before, we compute the spamicity of m_k by summing its spam contributions towards various products, w_2 weighted by $s(p_i)$ in (3).

$$s(m_k) = \sum_{i=1}^{|P|} w_2(m_k, p_i) s(p_i)\,;\, V_M = W_{MP} V_P \quad (3)$$

- Update p_i to reflect the extent it was spammed by members by summing the individual contribution of each member w_2, weighted by its spamicity.

$$s(p_i) = \sum_{k=1}^{|M|} w_2(m_k, p_i) s(m_k)\,;\, V_P = W_{MP}^T V_M \quad (4)$$

GSRank: Group Spam Rank

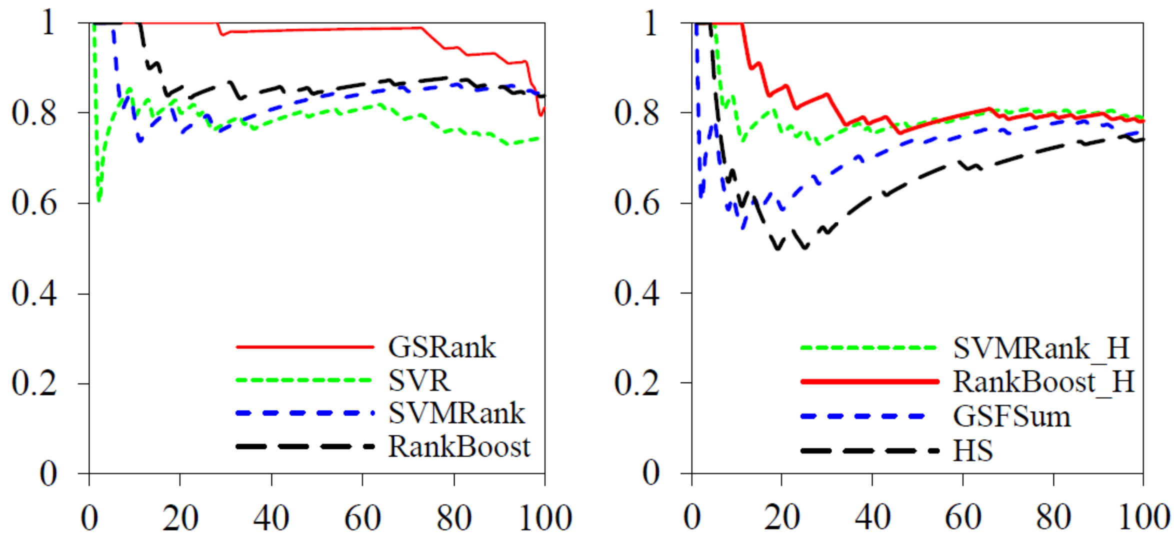
- Modeling collusion behaviors via GSRank
- GSRank: Group ↔ member ↔ product reinforcement based ranking [Mukherjee et al., WWW 2012]
- **Theoretical Guarantees:**
  - Lemma 1: GSRank is an <u>instance of an eigenvalue problem</u>
  - Theorem 1: GSRank converges

Algorithm: GSRank

Input: Weight matrices W_{PG}, W_{MP}, and W_{GM}
Output: Ranked list of candidate spam groups

1. Initialize $V_G^0 \leftarrow [0.5]_{|G|}$; $t \leftarrow 1$;
2. Iterate:
   i. $V_P \leftarrow W_{PG} V_G^{(t-1)}$; $V_M \leftarrow W_{MP} V_P$;
   ii. $V_G \leftarrow W_{GM} V_M$; $V_M \leftarrow W_{GM}^T V_G$;
   iii. $V_P \leftarrow W_{MP}^T V_M$; $V_G^{(t)} \leftarrow W_{PG}^T V_P$;
   iv. $V_G^{(t)} \leftarrow V_G^{(t)} / \| V_G^{(t)} \|_1$;
   until $\| V_G^{(t)} - V_G^{(t-1)} \|_\infty < \delta$
3. Output the ranked list of groups in descending order of V_G^*

GSRank Performance Evaluation



GSRank Performance Evaluation

Baselines: SVM, LR
Metric: AUC.

Feature sets:
GSF: Group spam features [Mukherjee et al., WWW 2011]

ISF: Indiv. spam features [Lim et al., CIKM 2010]

LF: Linguistic features [Ott et al., 2011]

Feature Settings	SVM	LR	SVR	SVM Rank	Rank Boost	SVM Rank_H	Rank Boost_H	GS Rank
GSF	0.81	0.77	0.83	0.83	0.85	0.81	0.83	**0.93**
ISF	0.67	0.67	0.71	0.70	0.74	0.68	0.72	
LF	0.65	0.62	0.63	0.67	0.72	0.64	0.71	
GSF + ISF + LF	0.84	0.81	0.85	0.84	**0.86**	0.83	0.85	

(a) The spamicity threshold of $\xi = 0.5$

Feature Settings	SVM	LR	SVR	SVM Rank	Rank Boost	SVM Rank_H	Rank Boost_H	GS Rank
GSF	0.83	0.79	0.84	0.85	0.87	0.83	0.85	**0.95**
ISF	0.68	0.68	0.73	0.71	0.75	0.70	0.74	
LF	0.66	0.62	0.67	0.69	0.74	0.68	0.73	
GSF + ISF + LF	0.86	0.83	0.86	0.86	**0.88**	0.84	0.86	

(b) The spamicity threshold of $\xi = 0.7$

Table 1: AUC results of different algorithms and feature sets.
All the improvements of GSRank over other methods are statistically significant at the confidence level of 95% based on paired *t*-test.

Group Opinion Spam on Amazon.com

1 of 1 people found the following review helpful: Practically FREE music, December 4, 2004 This review is from: Audio Xtract (CD-ROM) I can't believe for $10 (after rebate) I got a program that gets me free unlimited music. I was hoping it did half what was ….	2 of 2 people found the following review helpful: Like a tape recorder…, December 8, 2004 This review is from: Audio Xtract (CD-ROM) This software really rocks. I can set the program to record music all day long and just let it go. I come home and my ….	Wow, internet music! …, December 4, 2004 This review is from: Audio Xtract (CD-ROM) I looked forever for a way to record internet music. My way took a long time and many steps (frustrtaing). Then I found Audio Xtract. With more than 3,000 songs downloaded in …
3 of 8 people found the following review helpful: Yes – it really works, December 4, 2004 This review is from: Audio Xtract Pro (CD-ROM) See my review for Audio Xtract - this PRO is even better. This is the solution I've been looking for. After buying iTunes, ….	3 of 10 people found the following review helpful: This is even better than…, December 8, 2004 This review is from: Audio Xtract Pro (CD-ROM) Let me tell you, this has to be one of the coolest products ever on the market. Record 8 internet radio stations at once, ….	2 of 9 people found the following review helpful: Best music just got …, December 4, 2004 This review is from: Audio Xtract Pro (CD-ROM) The other day I upgraded to this TOP NOTCH product. Everyone who loves music needs to get it from Internet ….
5 of 5 people found the following review helpful: My kids love it, December 4, 2004 This review is from: Pond Aquarium 3D Deluxe Edition This was a bargain at $20 - better than the other ones that have no above water scenes. My kids get a kick out of the ….	5 of 5 people found the following review helpful: For the price you…, December 8, 2004 This review is from: Pond Aquarium 3D Deluxe Edition This is one of the coolest screensavers I have ever seen, the fish move realistically, the environments look real, and the ….	3 of 3 people found the following review helpful: Cool, looks great…, December 4, 2004 This review is from: Pond Aquarium 3D Deluxe Edition We have this set up on the PC at home and it looks GREAT. The fish and the scenes are really neat. Friends and family ….

Figure 1: Big John's Profile — Figure 2: Cletus' Profile — Figure 3: Jake's Profile

- All reviewed same 3 products giving all 5-stars + 100% helpfulness votes!.
- All reviews posted in a time window of 4 days
- Each only reviewed those 3 products,
- **Unlikely to be coincidental- Something seems fishy!**

Group Spam Variants

- Group detection via Network Footprints [Ye and Akoglu, PKDD 2015]
- A two-step approach:
- Step 1: Compute a Network Footprint Score (NFS) Vector of reviewers using Neighbor diversity and self-similarity per product to asses whether it is under attack by a spam campaign.
- Step 2: Cluster product vectors to find potential reviewer groups (e.g., via LSH [Gionis et al., VLDB 1999])

Outline

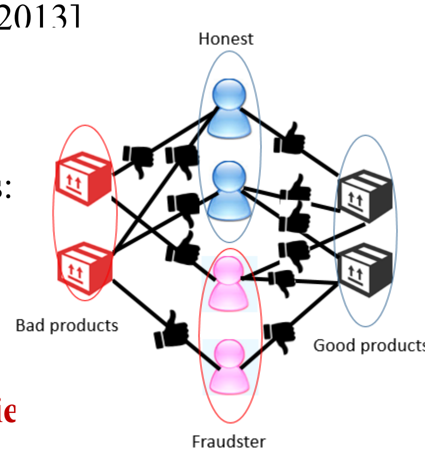
- ❑ Approach#1: Leveraging Linguistic Signals
- ❑ Approach#2: Behavioral Modeling
  - **P.1: Exploiting Anomalous Rating and Reviewing Behaviors**
  - **P.2: Modeling Group Spam/Collusion**
  - **P.3: Graph Based Methods**
  - **P.4: Distributional Methods**
- ❑ Approach#3: Statistical Modeling
- ❑ …

Leveraging User Product Review Networks

- ❑ Previous methods relied on use labeled data [Ott et al., ACL 2011; Mukherjee et al., WWW 2011, WWW 2012]
- ❑ Spamicity labels are prone to noise
- ❑ **Q: Can we learn spamicity labels automatically? (i.e., unsupervised learning)?**
- ❑ **For e.g., by leveraging different characteristics of data?**

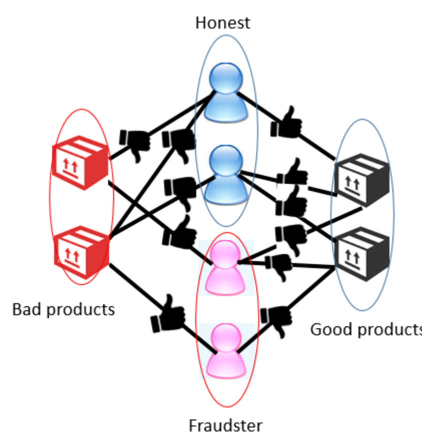
User Product Review Networks

- Product Review Networks [Akoglu et al., ICWSM 2013]
  - Input: (1) user-product review network
    (2) review sign (+:thumbs up/-:thumbs down)
  - Output: Classify objects to types specific categories:
    users: `honest' / `fraudster'
    products: `good' / `bad'
    reviews: `genuine' / `fake'
- **Automatically label the users, products, and revie network in the absence of meta-data (e.g., timestamp, review text)**



Spamicity Learning via User Product Review Networks

- FraudEagle [Akoglu et al., ICWSM 2013]
- Markov Random Field Formulation:
  - Signed network $G = (V, E)$, where $V = U \cup P$
    $U = \{u_1 \dots u_n\}; P = \{p_1 \dots p_n\}$
  - $E = \{e\}; e(u_i, p_j, s) s \in \{+, -\}$ (signed review links)
  - Each node in V and edge in E are random variables
    taking values from : $L_U = \{honest, fake\}, L_P = \{good, bad\}$
    $L_\epsilon = \{real, fake\}$



Spamicity Learning via User Product Review Networks

- FraudEagle [Akoglu et al., ICWSM 2013]
- Given an assignment y to all the unobserved variables Y^V and x to observed ones X^V, the model probability is:

Signed edges

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{Y_i \in \mathcal{Y}^{\mathcal{V}}} \phi_i(y_i) \prod_{e(Y_i^{\mathcal{U}}, Y_j^{\mathcal{P}}, s) \in \mathcal{E}} \psi_{ij}^{s}(y_i, y_j)$$

node labels as random variables

compatibility potentials

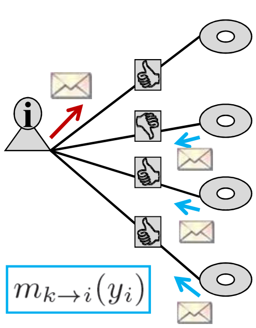
$$\phi_i^{\mathcal{U}}(y_i) = \psi_i^{\mathcal{U}}(y_i) \prod\nolimits_{e(Y_i^{\mathcal{U}}, X_j^{\mathcal{P}}, s) \in \mathcal{E}} \psi_{ij}^{s}(y_i)$$

prior belief

observed neighbor potentials

Signed Inference in MRFs

- Inference applied to signed bi-partite graphs
- Exact inference is NP-hard
- Approximate inference via Loopy Belief Propagation



I) Repeat for each node:

$$m_{i\to j}(y_j) = \alpha_1 \sum_{y_i \in \mathcal{L}^{\mathcal{U}}} \psi_{ij}^{s}(y_i, y_j)\, \phi_i^{\mathcal{U}}(y_i) \prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y}^{\mathcal{P}} \setminus Y_j} m_{k\to i}(y_i) \quad \forall y_j \in \mathcal{L}^{\mathcal{P}}$$

User i's belief of product j's state (good/bad)

II) At convergence:

$$b_i(y_i) = \alpha_2 \phi_i^{\mathcal{U}}(y_i) \prod_{Y_j \in \mathcal{N}_i \cap \mathcal{Y}^{\mathcal{P}}} m_{j\to i}(y_i)$$

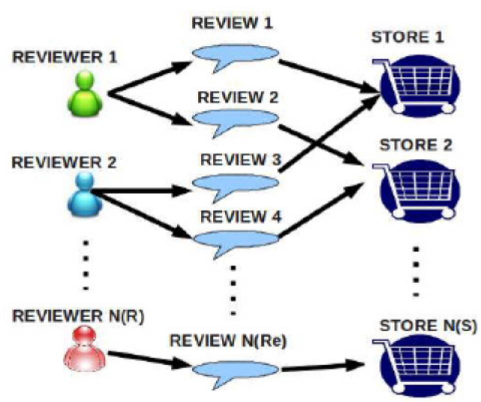
Belief of user i having label y_i

Graph Based Model Variants

- ❑ Online Store Spammer Detection [Wang et al., ICDM 2011]
- ❑ Reinforcement Ranking based on Graph Manifolds [Li et al., EMNLP 2013]

Modeling via Heterogeneous Graphs

- ❑ Online Store Spammer Detection [Wang et al., ICDM 2011]
- ❑ Heterogeneous graphs with three node types: (1) reviewer, (2) review, (3) store
- ❑ Introduces the notion of trustwothiness of reviewers, review honesty, and store reliability.



† Contains content originally appearing in [Wang et al., ICDM 2011]

Mutual Trust, Honesty, Agreement Relations

- Online Store Spammer Detection [Wang et al., ICDM 2011]
- Heterogeneous graphs with three node types: (1) reviewer, (2) review, (3) store
- Captures interrelationships:
  - Several honest review → more trustworthy reviewer
  - More +ve reviews form honest reviewer → reliable store
  - Review ratings supported/agreed by others → honest review

Semi-Supervised Ranking via Graph Manifolds

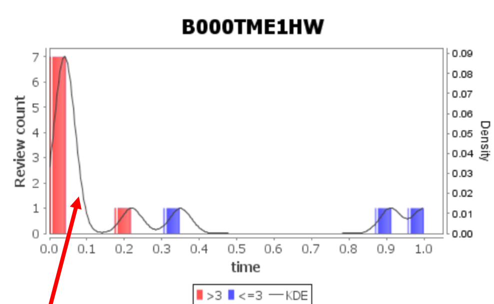
- Reinforcement Ranking based on Graph Manifolds [Li et al., EMNLP 2013]
- Semi-supervised manifold ranking relies on a small set of labeled individual reviews for training.
- Mutual Reinforcement Model on 3-layer Graph: $V_H = \{H_{i=1\ldots N_H}\}$; $V_R = \{R_{i=1\ldots H_R}\}$; $T = \{T_{i=1\ldots V}\}$
- Bootstrap using labeled data and then update score vectors using similarity functions

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
- ❑ Approach#2: Behavioral Modeling
  - **P.1: Exploiting Anomalous Rating and Reviewing Behaviors**
  - **P.2: Modeling Group Spam/Collusion**
  - **P.3: Graph Based Methods**
  - **P.4: Distributional Methods**
- ❑ Approach#3: Statistical Modeling
- ❑ …

Detecting Review Bursts

- ❑ Problem: Singleton review spam detection
- ❑ Exploit the burstiness nature of reviews to identify review spammers. [Fei et al. ICWSM 2013]
- ❑ Divide the life span of a product into bins.
- ❑ Fit histograms via Kernel Density Estimation, choose burst thresholds and count the number of reviews within each bin.



Early spamming reviewing burstiness detected for an entity

Detecting Review Bursts

- Modeling Review Bursts [Fei et al. ICWSM 2013]
- Model the reviewers in bursts and their co-occurrences in the same burst as a Markov Random Field. Reviewer states (spam/non-spam) are latent
- Co-occurrence of two reviewers in the same burst is represented by an edge connecting their corresponding hidden nodes.

$$\mathbf{b}_i(\tau) = k \prod_{j \in N(i)} \mathbf{m}_{ji}(\tau) \qquad \mathbf{m}_{ij}(\tau) = \sum_{\tau'} \psi(\tau, \tau') \prod_{n \in N(i) \setminus j} \mathbf{m}_{ni}(\tau')$$

Detecting Review Bursts

- Modeling Review Bursts [Fei et al. ICWSM 2013]
- Model the reviewers in bursts and their co-occurrences in the same burst as a Markov Random Field. Reviewer states (spam/non-spam) are latent
- Co-occurrence of two reviewers in the same burst is represented by an edge connecting their corresponding hidden nodes.

$$\mathbf{b}_i(\tau) = k \prod_{j \in N(i)} \mathbf{m}_{ji}(\tau) \qquad \mathbf{m}_{ij}(\tau) = \sum_{\tau'} \psi(\tau, \tau') \prod_{n \in N(i) \setminus j} \mathbf{m}_{ni}(\tau')$$

Ratio of Amazon verified purchase of a reviewer as state prior. Induce the overall spamming indicator of a reviewer in the LBP framework as local observation.

$$\mathbf{b}_i(\tau) = k\phi(\tau, \omega_i) \prod_{j \in N(i)} \mathbf{m}_{ji}(\tau)$$

$$\mathbf{m}_{ij}(\tau) = \sum_{\tau'} \psi(\tau, \tau')\phi(\tau', \omega_i) \prod_{n \in N(i) \setminus j} \mathbf{m}_{ni}(\tau')$$

Use Amazon Verified Purchase Tags as State Priors

Rating Distributional Analyses

- Q: Can we detect opinion spammers using distribution of review rating scores?
- Detect opinion spammers via divergence of rating distributions [Feng et al., ICWSM 2012]

Rating Distributional Analyses

- Hypothesis: Deceptive business entity that hires people to write fake reviews will necessarily distort its distribution of review scores, leaving distributional footprints behind. [Feng et al., ICWSM 2012]
- Analyses on 4 years of TripAdvisor data [2007-2011] revealed
- Significant increase in 4,5 star ratings over time – as if all hotels are consistently improving there services!

Time-Series Variants

- **Q: How to detect spammers who wrote fake reviews exactly once?**
- Not enough context to model user/review
- Singleton review spam detection

Time-Series, Distribution Analysis Variants

- Singleton review spam detection [Xie et al. KDD 2012]
- Hypothesis: truthful re-viewers' arrival pattern is stable and uncorrelated to their rating pattern temporally. In contrast, spam attacks are usually bursty and either positively or negatively correlated to the rating.

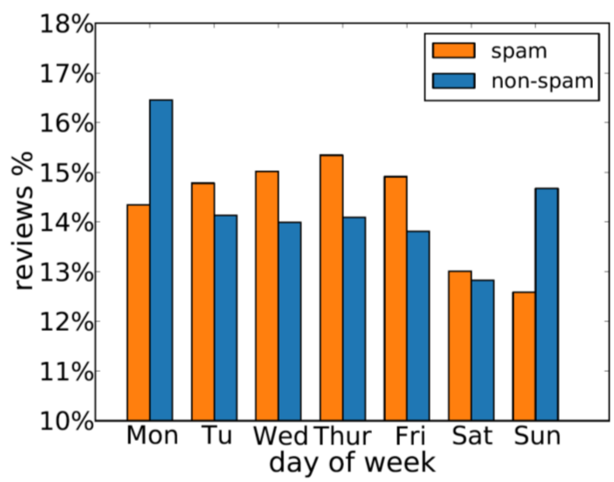
Spatio-Temporal Analysis on Large-Scale Data

- Q: What are the spatio-temporal dynamics of opinion spamming?
- Experiments on industry-scale filtered fake reviews – Dianping.com (Chinese Yelp) [Li et al., ICWSM 2015]
- Data Volume: over 6 Million reviews
- Richness: IP address and cookie information of users

大众点评
dianping.com

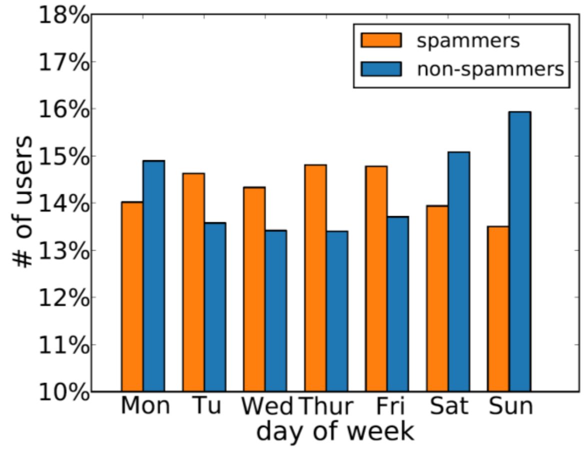
Temporal Patterns

- Authentic reviews are much more than fake reviews on Mondays and Sundays because they are based on real experience of dinners at weekends

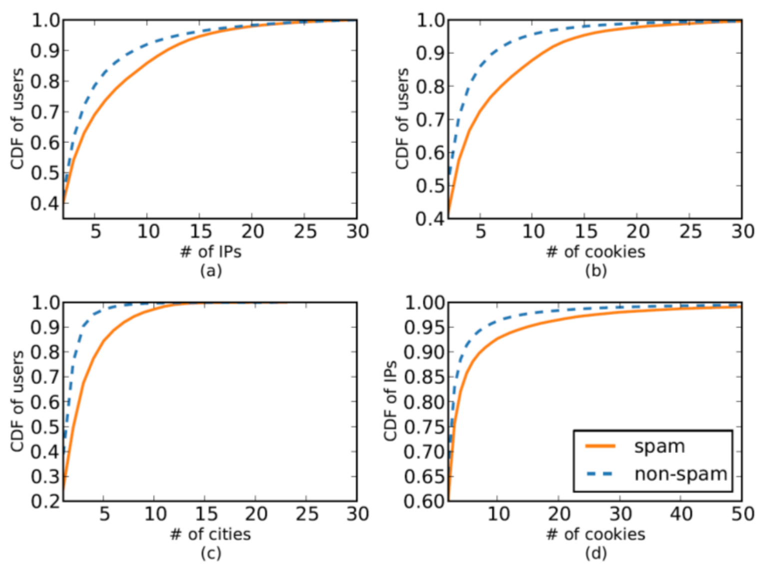


Temporal Patterns

- More non-spammers are registered at weekends and Mondays



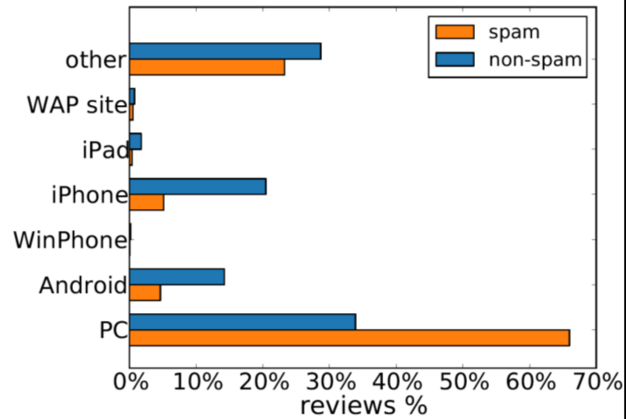
IP Address and Cookie Distributions



- Spammers switch IP addresses/cities and even browser cookies more often than ordinary/genuine users

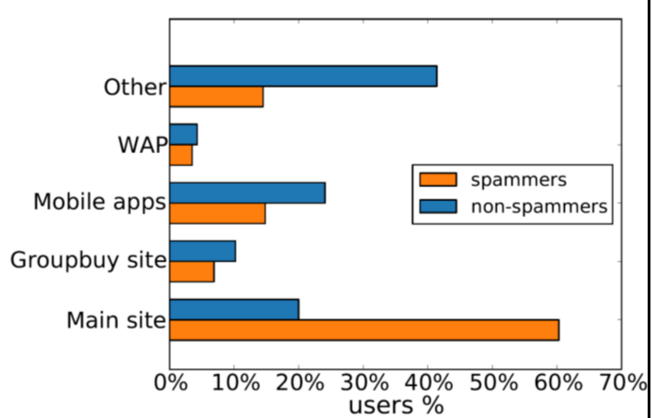
Reviewer Platform Distributions

- Review generation in main site is the fastest way, so spammers show a preference in registering and posting reviews on the main site



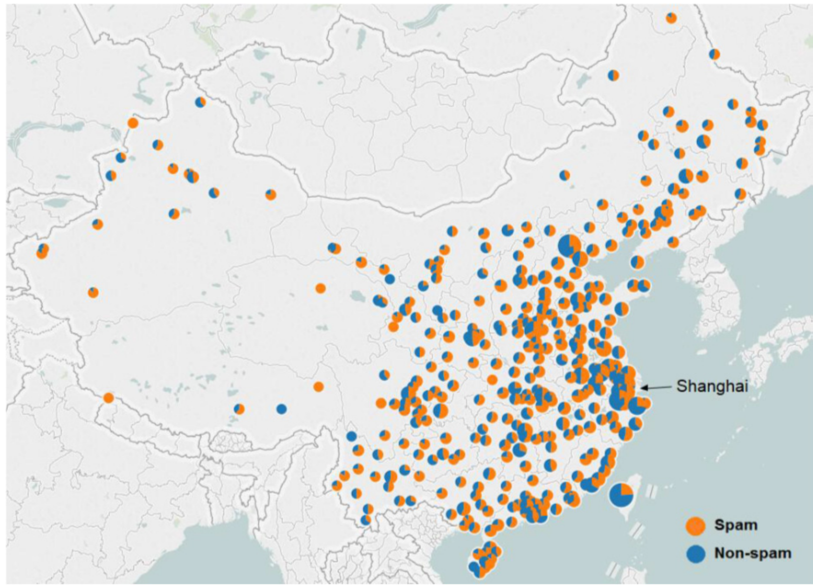
Reviewer Platform Distributions

- Reviews submitted via Mobile platforms/groupbuy/WAP sites tend to have lesser spammers (i.e., more truthful users)



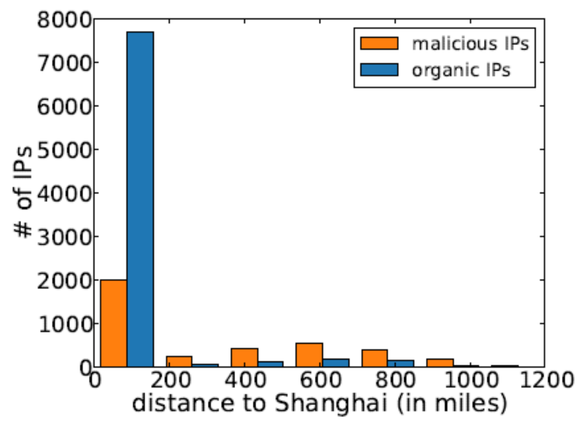
Opinion Spamming via Geo-Outsourcing

- People in large cities (a few big pie charts) are dominated by non-spammers (i.e., likely natives who actually ate in the restaurants)
- The further the cities are from Shanghai, the higher the ratios of spammers (spamming by hiring people who never visited Shanghai)



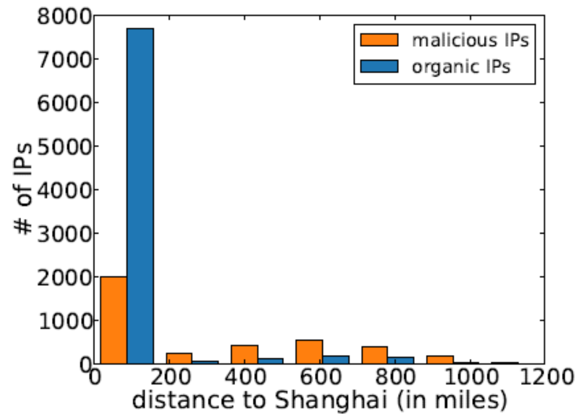
Geo-spatial IP distributions of Spammers

- Ratio of unique IPs of normal/spammers decrease with increase in distance from restaurant location.



Detecting Spammers via Abnormal Travel Speed

- Conjecture: As spamming is geo-outsourced, it is likely that professional spammers frequently change IP addresses to register many accounts in a short period of time
- Can we compute the avg. travel speed of reviewers?
- Reviewers with abnormal travels speeds indicate spammers.

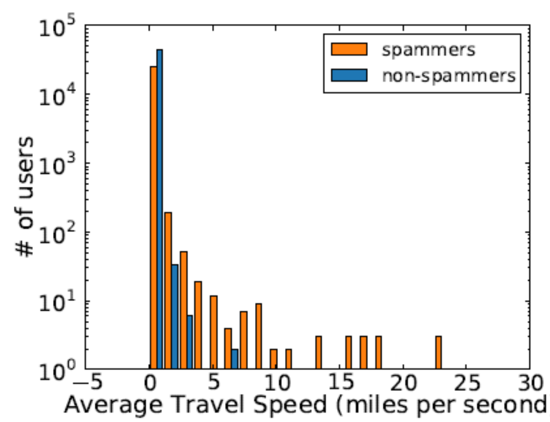


Detecting Spammers via Abnormal Travel Speed

- Define $S_u = \{r_1, \ldots r_{|S_u|}\}$ to be the sequence of reviews ordered by posting time-stamps.
- Compute Average Travel Speed (ATS) as (loc obtained from IP):

$$ATS_u = \frac{\sum_{i=2}^{|S_u|} distance(r_i.loc, r_{i-1}.loc)}{|S_u| - 1}$$

- ATS ∝ True Travel Speed as $|S_A| \propto$ $total\ time\ taken\ to\ generate\ |S_A| review$

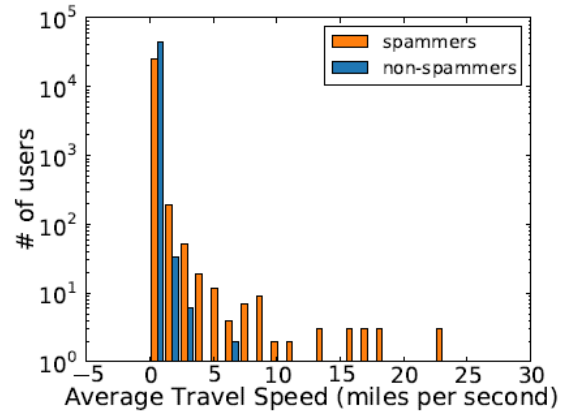


Detecting Spammers via Abnormal Travel Speed

- Define $S_u = \{r_1, \ldots r_{|S_u|}\}$ to be the sequence of reviews ordered by posting time-stamps.
- Compute Average Travel Speed (ATS) as:

$$ATS_u = \frac{\sum_{i=2}^{|S_u|} distance(r_i.loc, r_{i-1}.loc)}{|S_u| - 1}$$

- ATS ∝ True Travel Speed as $|S_A| \propto$ $total\ time\ taken\ to\ generate\ |S_A| review$
- Users with abnormal travels speeds are all spammers



Spatio-Temporal Features

- Novel spatio-temporal features explored in [Li et al., 2015]:

Feature	Description
regMainsite	Whether the user is registered on main site of Dianping
regTu2Tr	Whether the user is registered between Tue. and Thur
regDist2SH	Distance from the city where a user registered to Shanghai
ATS	Average Travel Speed
weekendPcnt	% of reviews written at weekends
pcPcnt	% of reviews posted through PC
avgDist2SH	Average distance from the city where the user posts each review to Shanghai

Spatio-Temporal Features

- Novel spatio-temporal features explored in [Li et al., 2015]:

Feature	Description
AARD	Average absolute rating deviation of users' reviews
uIPs	# of unique IPs used by the user
uCookies	# of unique cookies used by the user
uCities	# of unique cities where users write reviews

Spatio-Temporal Features

- Results of deception detection using spatio-temporal features explored in [Li et al., ICWSM 2015]

Method	Accuracy	Precision	Recall	F1
Unigram and Bigram	0.68	0.71	0.63	0.67
Behavioral Features	0.74	0.71	0.78	0.73
Proposed New Features	0.84	0.81	0.86	0.83
Combined	**0.85**	**0.83**	**0.87**	**0.85**

- Spatio-Temporal features alone are more effective than linguistic (n-gram) and behavioral features.

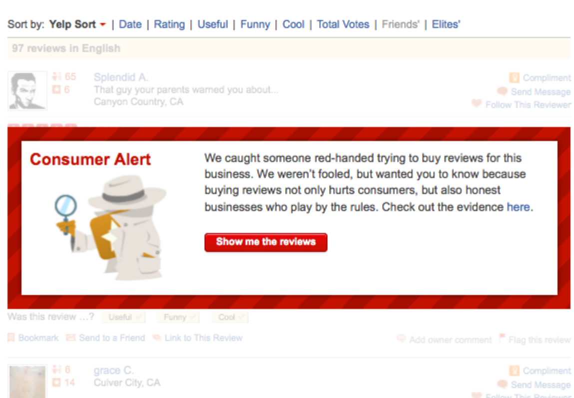
- Combining all linguistic, behavioral, and spatio-temporal features yield the best detection performance

Outline

- ❑ Approach#1: Leveraging Linguistic Signals
- ❑ Approach#2: Behavioral Modeling
  - **P.1: Exploiting Anomalous Rating and Reviewing Behaviors**
  - **P.2: Modeling Group Spam/Collusion**
  - **P.3: Graph Based Methods**
  - **P.4: Distributional Methods**
    - **Crowdsourced vs. Commercial Opinion Spamming**
- ❑ Approach#3: Statistical Modeling

Commercial Opinion Spam Filters: Case Study of Yelp

- ❑ Deception Research has mostly used duplicate reviews [Jindal and Liu, WSDM 2008] or AMT generated reviews [Ott et al., ACL 2011] as fakes.
- ❑ **Q: How does this compare to fake reviews detected by commercial filters? e.g., Yelp**



Commercial Opinion Spam Filters: Case Study of Yelp

- Deception Research has mostly used duplicate reviews [Jindal and Liu, 2008] or AMT generated reviews [Ott et al., 2011] as fakes.
- Q: How does this compare to fake reviews detected by commercial filters?
- Q: How much fake is out there?

From Yelp's official blog:

"about 25% of the reviews *submitted* to Yelp are not published on a business's listing"

\- Vince S., VP Communications & Public Affairs

Results of State-of-the-Art Supervised Learning

- AUC = 0.78 assuming duplicate reviews as fake [Jindal & Liu, WSDM 2008]. Duplicate reviews as fake is a naïve assumption.
- Accuracy = 90% using n-grams on Amazon Mechanical Turk (AMT) crowdsourced fake reviews [Ott et al., ACL 2011].

Q: How well do linguistic ngrams perform in detecting fake reviews filtered by Yelp? [Mukherjee et al., 2013]

Results of State-of-the-Art Supervised Learning

- AUC = 0.78 assuming duplicate reviews as fake [Jindal & Liu, WSDM 2008]. Duplicate reviews as fake is a naïve assumption.
- Accuracy = 90% using n-grams on Amazon Mechanical Turk (AMT) crowdsourced fake reviews [Ott et al., ACL 2011].

Q: How well do linguistic ngrams perform in detecting fake reviews filtered by Yelp? [Mukherjee et al., 2013]

Objective: Compare ngrams and behavioral features on Yelp data and postulate what might Yelp fake review filter be doing?

Applying Linguistic Features on Yelp Data

- Yelp Data Statistics
- Linguistic feature families:
- ngrmas, LIWC [Ott et al., 2011]
- Deep syntax: lexicalized and un-lexicalized production rules involving immediate or grandparent nodes of sentence parse trees [Feng et al., 2012]
- POS sequential patterns [Mukherjee and Liu, 2012]

Domain	fake	non-fake	% fake	total # reviews	# reviewers
Hotel	802	4876	14.1%	5678	5124
Restaurant	8368	50149	14.3%	58517	35593

Classification Experiments on Yelp Data

- SVM 5-fold CV results across different sets of features.
- WU → word unigram
- WB → word bigrams
- Top k% refers to using top features according to Information Gain (IG)

Features	P	R	F1	A		P	R	F1	A
Word unigrams (WU)	62.9	76.6	68.9	65.6		64.3	76.3	69.7	66.9
WU + IG (top 1%)	61.7	76.4	68.4	64.4		64.0	75.9	69.4	66.2
WU + IG (top 2%)	62.4	76.7	68.8	64.9		64.1	76.1	69.5	66.5
Word-Bigrams (WB)	61.1	79.9	69.2	64.4		64.5	79.3	71.1	**67.8**
WB+LIWC	61.6	69.1	69.1	64.4		64.6	79.4	71.0	**67.8**
POS Unigrams	56.0	69.8	62.1	57.2		59.5	70.3	64.5	55.6
WB + POS Bigrams	63.2	73.4	67.9	64.6		65.1	72.4	68.6	68.1
WB + Deep Syntax	62.3	74.1	67.7	64.1		65.8	73.8	69.6	67.6
WB + POS Seq. Pat.	63.4	74.5	68.5	64.5		66.2	74.2	69.9	67.7

Hotel Domain | Restaurant Domain

Classification Experiments on Yelp Data

- Across both hotel and restaurant domains, word unigrams only yield about 66% accuracy on real-life fake review data

Features	P	R	F1	A		P	R	F1	A
Word unigrams (WU)	62.9	76.6	68.9	65.6		64.3	76.3	69.7	66.9
WU + IG (top 1%)	61.7	76.4	68.4	64.4		64.0	75.9	69.4	66.2
WU + IG (top 2%)	62.4	76.7	68.8	64.9		64.1	76.1	69.5	66.5
Word-Bigrams (WB)	61.1	79.9	69.2	64.4		64.5	79.3	71.1	**67.8**
WB+LIWC	61.6	69.1	69.1	64.4		64.6	79.4	71.0	**67.8**
POS Unigrams	56.0	69.8	62.1	57.2		59.5	70.3	64.5	55.6
WB + POS Bigrams	63.2	73.4	67.9	64.6		65.1	72.4	68.6	68.1
WB + Deep Syntax	62.3	74.1	67.7	64.1		65.8	73.8	69.6	67.6
WB + POS Seq. Pat.	63.4	74.5	68.5	64.5		66.2	74.2	69.9	67.7

Hotel Domain | Restaurant Domain

Classification Experiments on Yelp Data

- Applying ngram classifiers on AMT generated fake reviews and Yelp filtered fake reviews

(a) Ott et al., (2011)

n	P	R	F1	A
Uni	86.7	89.4	88.0	87.8
Bi	88.9	89.9	89.3	88.8

(b) Hotel

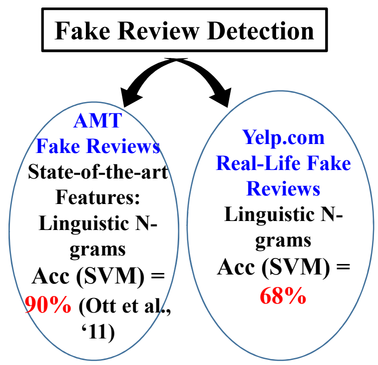
	P	R	F1	A
Uni	65.1	78.1	71.0	67.6
Bi	61.1	82.4	70.2	64.9

(c) Restaurant

	P	R	F1	A
Uni	65.1	77.4	70.7	67.9
Bi	64.9	81.2	72.1	68.5

Table 4: Comparison with Ott et al. (2011) based on SVM 5-fold CV results. Feature Set: Uni: Word unigrams, Bi: Word bigrams.

- AMT crafted fake reviews seem easier to detect than commercial fake reviews



- **Q: What causes this large accuracy difference?**

Word Distribution Analyses

- Compute word probability distributions in fake and non-fake reviews using Good-Turing smoothed unigram language models
- F: *Language model for fakes*
- N: *Language model for non − fakes*
- $KL(F||N) = \sum_i F(i)\, log_2\, (\frac{F(i)}{N(i)})$
- $F(i)$ and $N(i)$ are the respective probabilities of word i in fake and non-fake reviews

As word distributions govern the hardness of classification, relative word distributions in the language models of fakes and non-fake reviews can explain the difference in accuracy

Linguistic Divergence

- **Q: How much do fake reviews differ from non-fake reviews?**
- $KL(F||N)$ provides a quantitative measure
- But, $L(F||N)$ being assymetric, we also need JS-Div and $\Delta KL = KL(F||N) - KL(N||F)$

Dataset	# Terms	$KL(F\|N)$	$KL(N\|F)$	ΔKL	JS-Div
Ott et al. (2001)	6473	1.007	1.104	-0.097	0.274
Hotel	24780	2.228	0.392	1.836	0.118
Restaurant	80067	1.228	0.196	1.032	0.096
Hotel 4-5 ★	17921	2.061	0.928	1.133	0.125
Restaurant 4-5 ★	68364	1.606	0.564	1.042	0.105

Table 5: KL-Divergence of unigram language models.

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Observation 1: For the AMT data (Table 5, row 1), we get $KL(F||N) \approx KL(N||F)$ and $\Delta KL \approx 0$. However, for Yelp data (Table 5, rows 2-5), there are major differences, $KL(F||N) > KL(N||F)$ and $\Delta KL > 1$

Linguistic Divergence

- **Q: How much do fake reviews differ from non-fake reviews?**
- $KL(F||N)$ provides a quantitative measure
- But, $L(F||N)$ being assymetric, we also need JS-Div and $\Delta KL = KL(F||N) - KL(N||F)$

Dataset	# Terms	$KL(F\|N)$	$KL(N\|F)$	ΔKL	JS-Div
Ott et al. (2001)	6473	1.007	1.104	-0.097	0.274
Hotel	24780	2.228	0.392	1.836	0.118
Restaurant	80067	1.228	0.196	1.032	0.096
Hotel 4-5 ★	17921	2.061	0.928	1.133	0.125
Restaurant 4-5 ★	68364	1.606	0.564	1.042	0.105

Table 5: KL-Divergence of unigram language models.

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

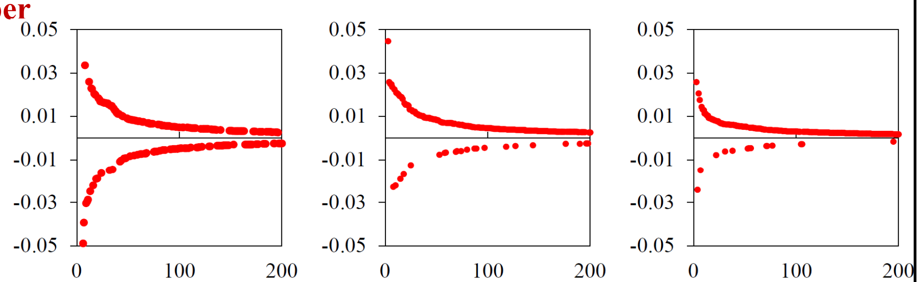
(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Observation 2: The JS-Div. of fake and non-fake word distributions in the AMT data is much larger (almost double) than Yelp data. This might be one reason why the AMT data is easier to classify…

Word Contributions to Linguistic Divergence

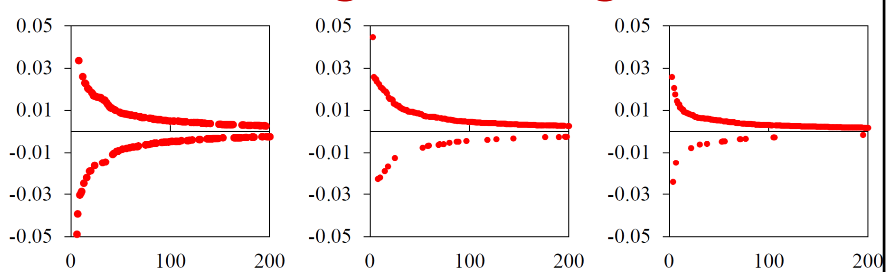
- **Q: How to measure per word contributions?**
- Define $\Delta KL^i_{Word} = KL_{Word}(F_i||N_i) - KL_{Word}(N_i||F_i)$; $KL_{Word}(F_i||N_i) = F(i)\log_2\left(\frac{F(i)}{N(i)}\right)$ and analogously $KL_{Word}(N_i||F_i)$
- Plot ΔKL^i_{Word} for top words in descending order of $\left|\Delta KL^i_{Word}\right|$ for various datasets.



(a) Ott et al. (b) Hotel (c) Restaurant

Word Contributions to Linguistic Divergence

- Define $\Delta KL^i_{Word} = KL_{Word}(F_i||N_i) - KL_{Word}(N_i||F_i)$, where $KL_{Word}(F_i||N_i) = F(i)\log_2\left(\frac{F(i)}{N(i)}\right)$ and analogously $KL_{Word}(N_i||F_i)$
- Plot ΔKL^i_{Word} for top words in descending order of $\left|\Delta KL^i_{Word}\right|$ for various datasets.
- Also compute the contribution of top k words to ΔKL for $k = 200$ and $k = 300$ in Table 6.



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

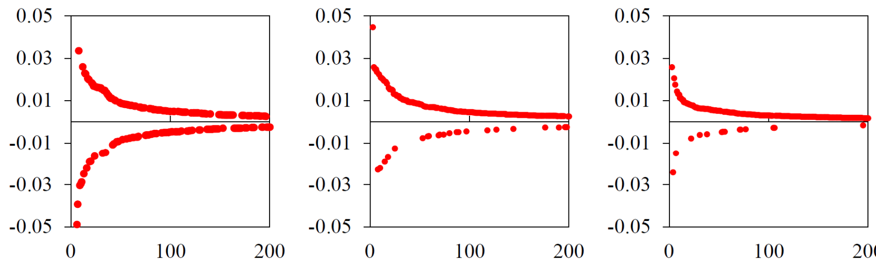
Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Turker's Language Model differs from Web Reviews

Observations (1):
AMT data has a "symmetric" distribution of two sets of words: i) E, appearing in fake with higher probability than non-fake ($F(i) > N(i)$, $\Delta KL_{Word}^{i \in E} > 0$) and (ii) G, appearing more in non-fake than fake ($N(i)F(i) > N(i)$, $\Delta KL_{Word}^{i \in E} > 0 > F(i)$ resulting in $\Delta KL_{Word}^{i \in F} < 0$)
Also Upper (y>0) and lower (y<0) curves are equally dense. $|E| \approx |G|$



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

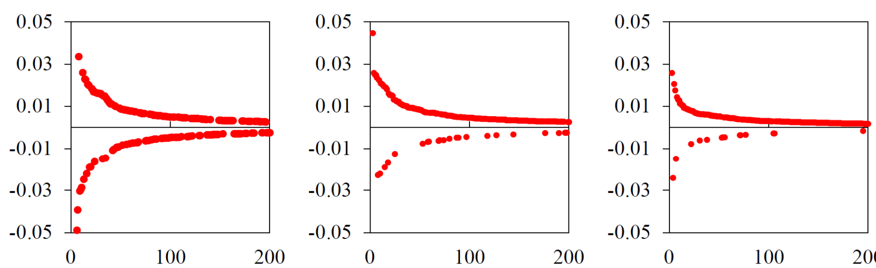
Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Turker's Language Model differs from Web Reviews

Observations (2):
Additionally, the top k = 200, 300 words (see Table 6(a, b), col. 1) only contribute about 20% to ΔKL for the AMT data. Thus, there are many more words in the AMT data having higher probabilities in fake than non-fake reviews (like those in set E) and also many words having higher probabilities in non-fake than fake reviews (like those in set G)



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

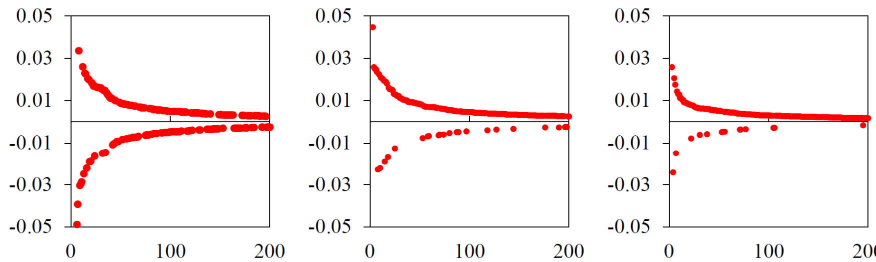
Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Turker Language Model is different from Web Reviews

Together Observations (1) and (2) tend to explain the high accuracy (90%) on AMT data



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

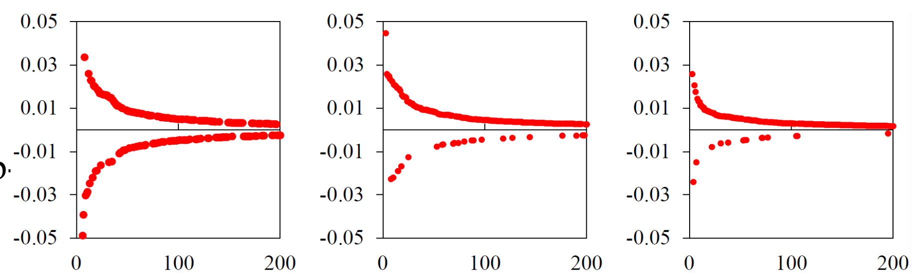
Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Are Yelp Spammers Smart?

- $KL(F||N)$ is much larger than $KL(N||F)$ and $\Delta KL > 1$. Fig. 1 (b-c) also show that among the top $k = 200$ words which contribute a major percentage ($\approx 70\%$) to ΔKL (see Table 6 (a))



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

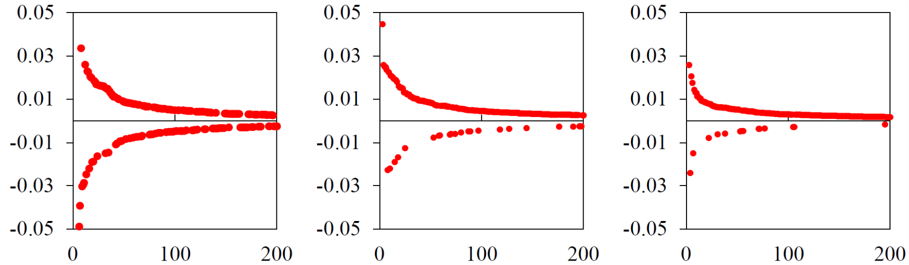
Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Are Yelp Spammers Smart?

- Thus, certain top k words contribute to ΔKL; and $\forall i \notin k$; $\Delta KL^{i}_{Word} \approx 0$



(a) Ott et al. (b) Hotel (c) Restaurant

% Contr.	Ott. et al.	Ho.	Re.	Ho. 4-5 ★	Re. 4-5★
ΔKL	20.1	74.9	70.1	69.8	70.3

(a) $k = 200$

Ott. et al.	Ho.	Re.	Ho. 4-5★	Re. 4-5★
22.8	77.6	73.1	70.7	72.8

(b) $k = 300$

Table 6: Percentage (%) of contribution to ΔKL for top k words across different dataset. Ho: Hotel and Re: Restaurant.

Are Yelp Spammers Smart?

Plausible inference:

Yelp spammers (authors of filtered reviews) made an effort (are smart enough) to ensure that their fake reviews align with non-fakes (i.e., have most words that also appear in truthful reviews) to sound "convincing".

However, during the process/act of "faking" they happened to *overuse* some words consequently resulting in much higher frequencies of certain words in their fake reviews than other non-fake reviews.

Deception Signals in Yelp Fake Reviews

- **Q: What are those words that spammers tend to overuse?**
- *us*, *price*, *stay*, *feel*, *nice*, *deal*, *comfort*, etc. in the hotel domain; and *options*, *went*, *seat*, *helpful*, *overall*, *serve*, *amount*, etc. in the restaurant domain.
- More use of personal pronouns and emotions → pretense (?) [Result later corroborated by Li et al., ACL 2014]
- **Q: How do these differ from traditional lies in deception?**

Word (w)	ΔKL_{Word}
us	0.0446
area	0.0257
price	0.0249
stay	0.0246
said	-0.0228
feel	0.0224
when	-0.0221
nice	0.0204
deal	0.0199
comfort	0.0188

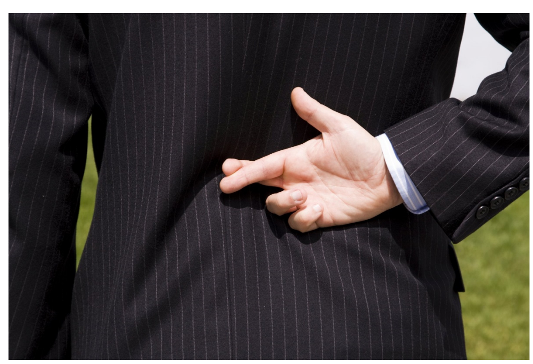
Hotel Domain

Word (w)	ΔKL_{Word}
places	0.0257
options	0.0130
evening	0.0102
went	0.0092
seat	0.0089
helpful	0.0088
overall	0.0085
serve	0.0081
itself	-0.0079
amount	0.0076

Restaurant Domain

Conventional Deception/Lying Signals

- Newman et al., [2003] reports lying/deception communication is characterized by the use of **fewer** first-person pronouns, more negative emotion words, and more motion/action words
- Fewer personal pronouns refers to the psycholinguistic process of “detachment” – liars trying to “disassociate” themselves [Knapp et al., 1974].



Vertical Deceptive Signals in Fake Reviews

- ❑ Psychological attachment: Fake reviews actually like to use more first-person pronouns such as I, myself, mine, we, us, etc., to make their reviews **sound more convincing** and to **give readers the impression that their reviews express their true experiences**

- ❑ "Attachment" is distinctly different from the psychological process of "detachment" which results in the use of fewer first-person pronouns like I, myself, mine, etc.

dreamypoo
New York City, New York
2 reviews
2 helpful votes

"Great for Prostitutes and Drug Dealers!!!"

Reviewed August 2, 2011

2 people found this review helpful

I can't fully express the experience I had at this... place in this review but I will try my best. Of course I booked the room because of the price. I have stayed at other hotels that were reasonably priced and decent so I was hoping for the same here. My Fiance and I were not looking for the Taj Mahal but dang!!! Check in took forever because the woman at the desk was training someone. They charged my credit card at check in which made me realize that the place must be awful if they have to do that! The courtyard that you have to walk through to get to your room looked like part welfare hotel part prison yard! Scary!!! The room was OLD and beat up! The desk was missing a drawer. There were stains on the filthy carpet. The shower had mold on the floor so we had to wear our flip flops when we took a shower. There were no amenities at all!!! No coffee pot, no iron or ironing board, no clock radio... NOTHING!!! When we went to the front desk for a clock they said it was a $10.00 and a $20.00 deposit to use the phone! Then.... our toilet became backed up that night and they did not fix it until noon the next day!!!!!!!!!!!!! The towels were dingy as were the sheets. I really can't believe there were no bed bugs! Whatever you do... stay somewhere else! The place is a rat hole!!!!

Stayed July 2011, traveled as a couple

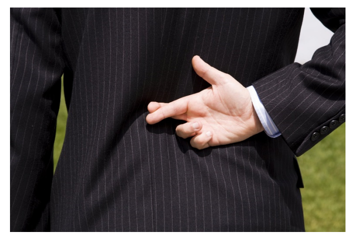
Value | Cleanliness
Sleep Quality | Service

Less

Was this review helpful? Yes

Problem with this review?

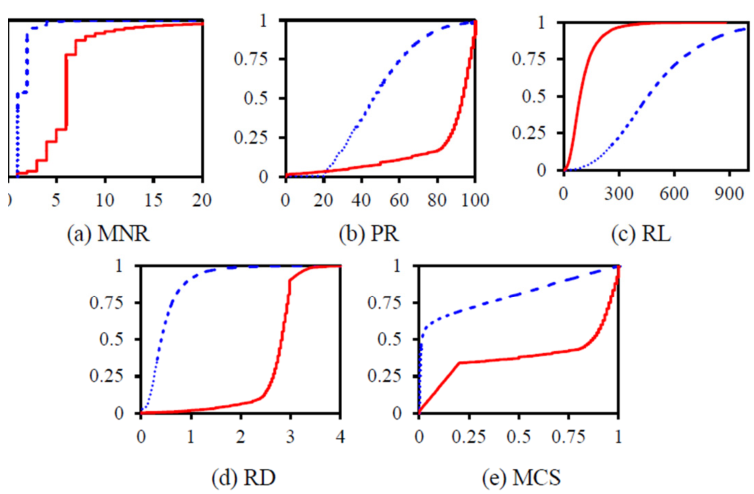
VS



Behavioral Patterns of Yelp Spammers

- ❑ Behavioral features:
  - ▪ MNR: Max No. of Reviews/day
  - ▪ PR: % of positive reviews
  - ▪ RL: Review length
  - ▪ RD: Rating Deviation
  - ▪ MCS: Max Content Similarity

- ❑ CDFs show sufficient gaps → Discriminative



e 2: CDF (Cumulative Distribution Function) of behavioral features. Cumulative percentage of spammers (in red/solid) and non-spammers (in blue/dotted) vs. behavioral feature value.

Linguistic vs. Behavioral Features

- **Q: Which feature families are more discriminative?**
- Behavioral features alone have a major gain
- Linguistic features further improves gains
- Linguistic features obtain > 50% accuracy → Contain subtle signals

Feature Setting	P	R	F1	A	P	R	F1	A
Unigrams	62.9	76.6	68.9	65.6	64.3	76.3	69.7	66.9
Bigrams	61.1	79.9	69.2	64.4	64.5	79.3	71.1	67.8
Behavior Feat.(BF)	81.9	84.6	**83.2**	**83.2**	82.1	87.9	**84.9**	**82.8**
Unigrams + BF	83.2	80.6	81.9	83.6	83.4	87.1	85.2	84.1
Bigrams + BF	86.7	82.5	**84.5**	**84.8**	84.1	87.3	**85.7**	**86.1**

Hotel Domain Restaurant Domain

What are the Most Discriminative Features?

- Ablation experiments to assess feature contribution
- Graceful degradation → Every feature contributes to some extent
- Dropping RL, MCS reduces accuracy by 4-6% → Potentially more discriminative

Feature Setting	P	R	F1	A
Unigrams	62.9	76.6	68.9	65.6
Bigrams	61.1	79.9	69.2	64.4
Behavior Feat.(BF)	81.9	84.6	**83.2**	**83.2**
Unigrams + BF	83.2	80.6	81.9	83.6
Bigrams + BF	86.7	82.5	**84.5**	**84.8**

(a): Hotel

P	R	F1	A
64.3	76.3	69.7	66.9
64.5	79.3	71.1	67.8
82.1	87.9	**84.9**	**82.8**
83.4	87.1	85.2	84.1
84.1	87.3	**85.7**	**86.1**

(b): Restaurant

Dropped Feature	P	R	F1	A
MNR	84.9	80.6	82.7	83.3
PR	82.9	78.2	80.5	80.1
RL	82.7	78.0	80.3	79.7
RD	85.2	81.6	83.4	84.0
MCS	83.9	80.1	81.9	82.9

(a): Hotel

P	R	F1	A
82.8	86.0	84.4	84.4
81.3	83.4	82.3	82.5
81.8	82.9	82.3	81.8
83.4	86.7	85.0	85.7
82.8	85.0	83.9	84.3

(b): Restaurant

Main Results on Yelp Data

- Amazon Mechanical Turk (AMT) crowdsourced fake reviews may not be representative of real-life fake reviews in commercial setting such as Yelp (acc. 90% vs. 68%).
- Yelp's fake review filter might be using a behavioral based approach.
- Abnormal behaviors strongly correlated ($p<0.01$) with Yelp's fake reviews =>Yelp filtering seems reliable

Outline

- Approach#2: Behavioral Modeling
- Approach#3: Statistical Modeling
  - **P.1: Latent Variable Models**
  - **P.2: Positive Unlabeled (PU) Learning**
  - **P.3: Collective PU Learning**
  - **P.4: Typed Markov Random Fields**
- …

Distributional Divergence

- **Hypothesis:** Spammers/Non-spammers differ in behavioral densities [Mukherjee et al., WWW 2011, WWW 2012, ICWSM 2013]
- **Q: How much do they differ?**

If quantified → Discover latent populations distributions from observed behavioral footprints

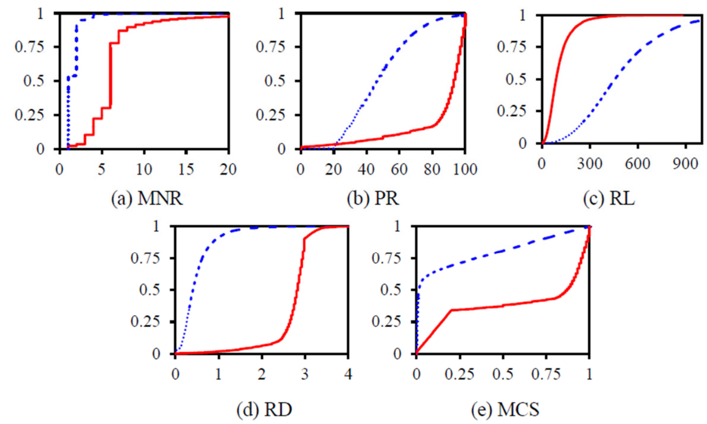


Figure 2: CDF (Cumulative Distribution Function) of behavioral features. Cumulative percentage of spammers (in red/solid) and non-spammers (in blue/dotted) vs. behavioral feature value.

Distributional Divergence

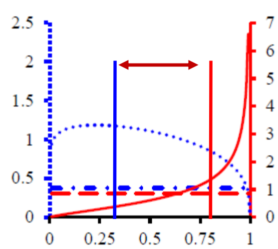
- **Model spamicity as latent** with other **observed behavioral footprints** [Mukherjee et al., KDD 2013]
- Modeling behaviors as Beta distributions:

(1) Extreme Rating (EXT)

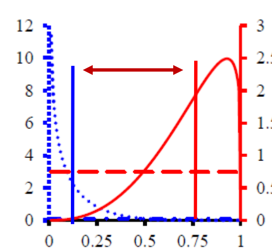
- $\theta_S^{EXT} \sim Beta\ (\alpha_1, \beta_1)$
- $\theta_N^{EXT} \sim Beta\ (\alpha'_1, \beta'_1)$

(2) Reviewing Burstiness (BST)

- $\psi_S^{BST} \sim Beta\ (\alpha_2, \beta_2)$
- $\psi_N^{BST} \sim Beta\ (\alpha'_2, \beta'_2)$



$\overline{\theta_S^{EXT}} = 0.86, \overline{\theta_N^{EXT}} = 0.36$



$\overline{\psi_S^{BST}} = 0.75, \overline{\psi_N^{BST}} = 0.10$

Distributional Divergence

- **Model spamicity as latent** with other **observed behavioral footprints** [Mukherjee et al., KDD 2013]
- Modeling behaviors as Beta distributions:

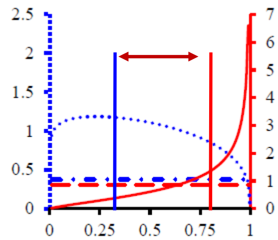
(1) Extreme Rating (EXT)

- $\theta_S^{EXT} \sim Beta\ (\alpha_1, \beta_1)$
- $\theta_N^{EXT} \sim Beta\ (\alpha_1', \beta_1')$

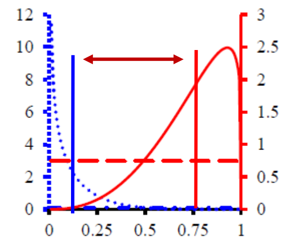
(2) Reviewing Burstiness (BST)

- $\psi_S^{BST} \sim Beta\ (\alpha_2, \beta_2)$
- $\psi_N^{BST} \sim Beta\ (\alpha_2', \beta_2')$

Deception Detection → Generative Model Based Clustering



$\overline{\theta_S^{EXT}} = 0.86, \overline{\theta_N^{EXT}} = 0.36$



$\overline{\psi_S^{BST}} = 0.75, \overline{\psi_N^{BST}} = 0.10$

Author Spamicity Model

- Latent Variables:
  - Author spamicity $s_a \sim Beta\ (\alpha)$
  - Class label (spam/non-spam) of a review, r, $\pi_r \in \{s, n\}$
- Latent Behavioral Distributions
  - Review behavioral models:
    - i. Rating Abuse, $\theta^{RA} \sim Beta\ (\gamma^{RA})$
    - ii. Duplicate Review Posting, $\theta^{DUP} \sim Beta\ (\gamma^{DUP})$
    - iii. Extreme Review Rating, $\theta^{EXT} \sim Beta\ (\gamma^{EXT})$
    - iv. Rating Deviation, $\theta^{DEV} \sim Beta\ (\gamma^{DEV})$
    - v. Early Time Frame, $\theta^{ETF} \sim Beta\ (\gamma^{ETF})$
  - Author behavioral models:
    - i. Content Similarity, $\theta^{CS} \sim Beta\ (\psi^{CS})$
    - ii. Max No. of Reviews, $\theta^{MNR} \sim Beta\ (\psi^{MNR})$
    - iii. Reviewing Burstiness, $\theta^{BST} \sim Beta\ (\psi^{BST})$
    - iv. Ratio of first reviews, $\theta^{RFR} \sim Beta\ (\psi^{RFR})$

Opinion spammers differ from others on behavioral dimensions resulting in a separation margin between distributions of two naturally occurring clusters: spammers and non-spammers

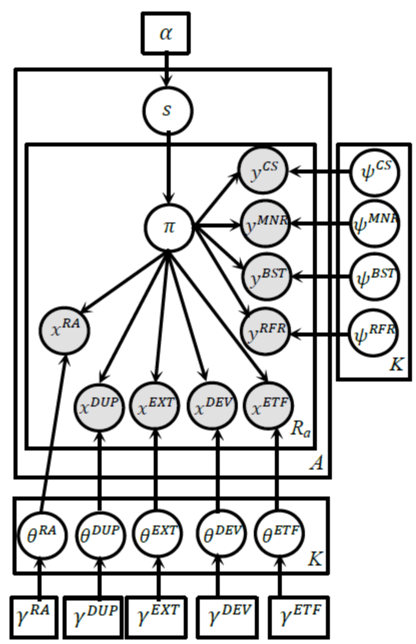
Author Spamicity Model

- Latent Variables:
  - Author spamicity $s_a \sim Beta\ (\alpha)$
  - Class label (spam/non-spam) of a review, r, $\pi_r \in \{s, n\}$
- Latent Behavioral Distributions
  - Review behavioral models:
    - i. Rating Abuse, $\theta^{RA} \sim Beta\ (\gamma^{RA})$
    - ii. Duplicate Review Posting, $\theta^{DUP} \sim Beta\ (\gamma^{DUP})$
    - iii. Extreme Review Rating, $\theta^{EXT} \sim Beta\ (\gamma^{EXT})$
    - iv. Rating Deviation, $\theta^{DEV} \sim Beta\ (\gamma^{DEV})$
    - v. Early Time Frame, $\theta^{ETF} \sim Beta\ (\gamma^{ETF})$
  - Author behavioral models:
    - i. Content Similarity, $\theta^{CS} \sim Beta\ (\psi^{CS})$
    - ii. Max No. of Reviews, $\theta^{MNR} \sim Beta\ (\psi^{MNR})$
    - iii. Reviewing Burstiness, $\theta^{BST} \sim Beta\ (\psi^{BST})$
    - iv. Ratio of first reviews, $\theta^{RFR} \sim Beta\ (\psi^{RFR})$

**Observed features: RA, DUP, EXT, …BST, RFR computed from Amazon.com review dataset of 50,704 reviewers, 985,765 reviews,
and 112,055 products.**

Generative Process

1. For each class/cluster, $k \in \{\hat{s}, \hat{n}\}$:
   Draw $\theta_k^{f \in \{DUP,\ldots,RA\}} \sim Beta(\gamma^f)$
2. For each (author), $a \in \{1 \ldots A\}$:
   i. Draw spamicity, $s_a \sim Beta(\alpha^a)$;
   ii. For each review, $r_a \in \{1 \ldots R_a\}$:
      a. Draw its class, $\pi_{r_a} \sim Bern(s_a)$
      b. Emit review features $f \in \{DUP, \ldots, RA\}$:
         $x_{r_a}^f \sim Bern\left(\theta_{\pi_{r_a}}^f\right)$;
      c. Emit author features $f \in \{CS, \ldots, RFR\}$:
         $y_{r_a}^f \sim \psi_{\pi_{r_a}}^f$;



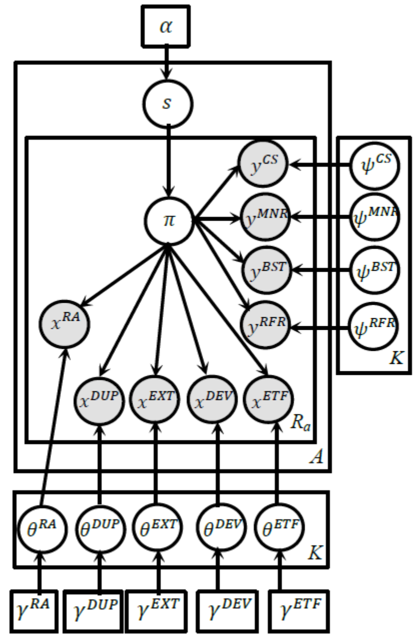
Inference

- Approximate posterior inference with Monte Carlo Gibbs sampling
- Rao-Blackwellization to reduce sampling variance by collapsing on LVs s and θ^f

$$p(\pi_i = k|\boldsymbol{\pi}_{\neg i} \ldots) \propto$$
$$\frac{n_{a,k_{\neg i}} + \alpha_k^a}{(n_a + \alpha_{\hat{s}}^a + \alpha_{\hat{n}}^a)_{\neg i}} \times \prod_{f \in \{DUP,EXT,DEV,ETF,RA\}} \left(g(f,k,x_{a,r}^f)\right) \times \prod_{f \in \{CS,MNR,BST,RFR\}} \left(p(y_{a,r}^f | \psi_{\pi_i}^f)\right)$$

$$g(f,k,x_{a,r}^f) = \begin{cases} \frac{(n_{k,P}^f + \gamma_k^f)_{\neg i}}{(n_k + \gamma_{\hat{s}}^f + \gamma_{\hat{n}}^f)_{\neg i}}, & if \;\; x_{a,r}^f = 1 \\ \frac{(n_{k,A}^f + \gamma_{\neg k}^f)_{\neg i}}{(n_k + \gamma_{\hat{s}}^f + \gamma_{\hat{n}}^f)_{\neg i}}, & if \;\; x_{a,r}^f = 0 \end{cases}$$

$$p(y_{a,r}^f | \psi_{\pi_i}^f) \propto (y_{a,r}^f)^{\psi_{i_1}^f - 1} (1 - y_{a,r}^f)^{\psi_{i_2}^f - 1}$$



Inference

Algorithm 1 Inference using MCMC Gibbs Sampling

1. Initialization:

Randomly assign review clusters, $\pi_{r_a} = \begin{cases} \hat{n}, z < 0.5 \\ \hat{s}, z \geq 0.5 \end{cases}; z \sim U(0,1)$

2. Iterate n = 1 to N_{max}: // $N_{max} = 3000$

For author, a = 1 to A:

For review r_a = 1 to R_a:

i. Flush cluster assignment, π_{r_a};

ii. Sample $\pi_{r_a} \sim p(\pi = k| \ldots)$ using (10);

iii. Update $n_{k,[]}^{f=DUP}$, $n_{k,[]}^{f=EXT}$, $n_{k,[]}^{f=DEV}$, $n_{k,[]}^{f=ETF}$, $n_{k,[]}^{f=RA}$ for $k \in \{\hat{s}, \hat{n}\}$

End for

End for

If $n > N_{Burn\_In}$: // $N_{BurnIn} = 250$

For author, a = 1 to A:

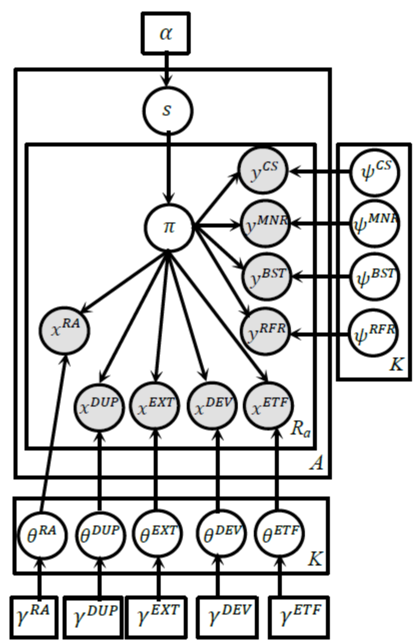
For review r_a = 1 to R_a:

Update $\psi_k^{f=CS}$, $\psi_k^{f=MNR}$, $\psi_k^{f=BST}$, $\psi_k^{f=RFR}$; $k \in \{\hat{s}, \hat{n}\}$ using (11)

End for

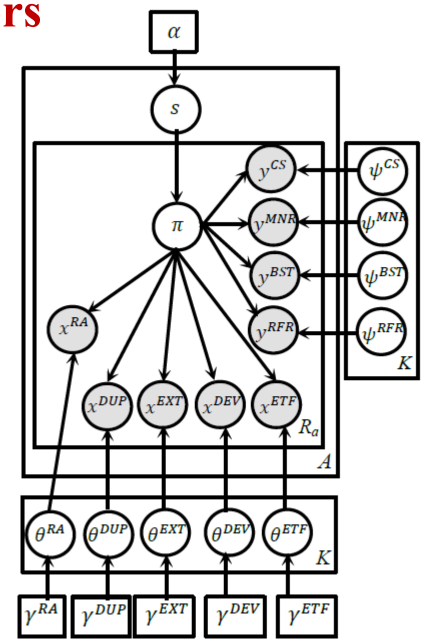
End for

End if



Author Spamicity Priors

- Algorithm 1 performs inference using uninformed priors (i.e., hyperparameters α and γ are set to (1,1)).
- Priors for author spamicity and latent review behaviors (α and γ) affect spam/non-spam cluster assignments
- Hence, posterior estimates of author spamicity can be improved if hyperparameters α and γ are estimated from the data.



Hyperparameter EM

- Estimate hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ that maximize the model's complete log-likelihood, L.
- Optimizer: L-BFGS

Algorithm 2 Single-sample Monte Carlo EM
1. Initialization:
Start with uninformed priors: $\alpha^a \leftarrow (1,1)$; $\gamma^f \leftarrow (1,1)$
2. Repeat:
i. Run Gibbs sampling to steady state (Algorithm 1) using current values of α^a, γ^f.
ii. Optimize α^a using (12) and γ^f using (14)
Until convergence of α^a, γ^f

$$\alpha_k^a = \operatorname{argmax}_{\alpha_k^a}\begin{pmatrix}\log\Gamma(\alpha_{\hat{s}}^a+\alpha_{\hat{n}}^a)+\log\Gamma(\alpha_{\hat{s}}^a+n_{a,\hat{s}})+\log\Gamma(\alpha_{\hat{n}}^a+n_{a,\hat{n}})\\ -\log\Gamma(\alpha_{\hat{s}}^a)-\log\Gamma(\alpha_{\hat{n}}^a)-\log\Gamma(n_a+\alpha_{\hat{s}}^a+\alpha_{\hat{n}}^a)\end{pmatrix} \quad (12)$$

$$\frac{\partial\mathcal{L}}{\partial\alpha_k^a} = \Psi(\alpha_{\hat{s}}^a+\alpha_{\hat{n}}^a)+\Psi(\alpha_k^a+n_{a,k})-\Psi(\alpha_k^a)-\Psi(n_a+\alpha_{\hat{s}}^a+\alpha_{\hat{n}}^a) \quad (13)$$

$$\gamma_k^f = \operatorname{argmax}_{\gamma_k^f}\begin{pmatrix}\log\Gamma(\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f)+\log\Gamma(\gamma_{\hat{s}}^f+n_{k,P}^f)+\log\Gamma(\gamma_{\hat{n}}^f+n_{k,A}^f)\\ -\log\Gamma(\gamma_{\hat{s}}^f)-\log\Gamma(\gamma_{\hat{n}}^f)-\log\Gamma(n_k+\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f)\end{pmatrix} \quad (14)$$

$$\frac{\partial\mathcal{L}}{\partial\gamma_{\hat{s}}^f} = \Psi(\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f)+\Psi(\gamma_{\hat{s}}^f+n_{\hat{s},P}^f)-\Psi(\gamma_{\hat{s}}^f)-\Psi(n_k+\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f) \quad (15)$$

$$\frac{\partial\mathcal{L}}{\partial\gamma_{\hat{n}}^f} = \Psi(\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f)+\Psi(\gamma_{\hat{n}}^f+n_{\hat{n},A}^f)-\Psi(\gamma_{\hat{n}}^f)-\Psi(n_k+\gamma_{\hat{s}}^f+\gamma_{\hat{n}}^f) \quad (16)$$

where, $\Psi(\cdot)$ denotes the digamma function.

Evaluating ASM via Review Classification

- If ASM is effective → it should rank highly likely spammers at the top and highly likely non-spammers at the bottom.
- So, supervised classification of reviews of likely spammers and likely non-spammers can the spamicity ranking.

This evaluation is based on the hypothesis that spam opinions can be separated from truthful ones using ngrams [Ott et al., 2011].

Review classification of top/bottom authors being good → ASM spamicity ranking of reviewers is effective because text classification concurs with the abnormal behavior spam detection of ASM

Evaluating ASM via Review Classification

- ASM outperforms various baselines:
  - Feature Sum
  - Helpfulness
  - SVMRank
  - RankBoost

Table 2 (a)

k	ASM-UP				ASM-IP				ASM-HE				SVMRank			
(%)	P	R	F1	A	P	R	F1	A	P	R	F1	A	P	R	F1	A
5	77.7	74.0	75.8	75.5	77.9	74.8	76.3	75.7	79.6	75.1	77.3	77.4	72.1	74.7	73.4	73.1
10	68.5	62.9	65.6	63.5	72.1	69.5	70.8	72.8	76.8	70.3	73.4	73.4	67.9	70.3	69.1	70.4
15	62.9	59.9	61.4	60.2	66.8	64.5	65.6	66.1	68.9	67.4	68.1	66.7	57.2	60.9	58.9	59.2

Table 2 (b)

k	RankBoost				FSum				HS			
(%)	P	R	F1	A	P	R	F1	A	P	R	F1	A
5	74.6	75.1	74.8	74.6	76.1	73.6	74.8	75.2	57.8	61.7	59.7	59.8
10	68.1	71.6	69.8	71.2	67.3	60.2	63.6	61.4	58.9	60.8	59.8	60.6
15	58.3	57.8	58.0	59.8	60.2	55.3	57.6	57.2	61.7	58.0	59.8	58.2

Table 2 (a, b): 5-fold SVM CV for review classification using top *k* (%) authors' reviews as the spam (+) class and bottom *k* % authors' reviews as the non-spam (-) class. P: Precision, R:Recall, F1:F1-Score, A:Accuray.

Evaluating ASM via Expert Evalaution

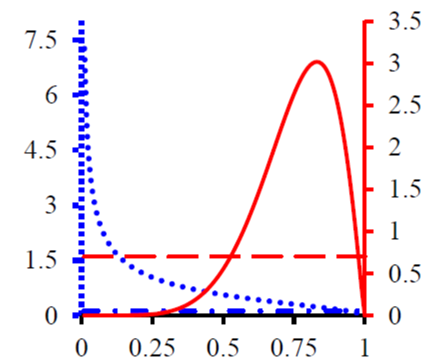
	ASM-UP			ASM-IP			ASM-HE			SVMRank			RankBoost			FSum			HS		
	B_1	B_2	B_3	B_1	B_2	B_3	B_1	B_2	B_3	B_1	B_2	B_3	B_1	B_2	B_3	B_1	B_2	B_3	B_1	B_2	B_3
J_1	31	15	3	36	11	1	43	5	0	36	19	1	37	13	1	34	13	0	6	14	17
J_2	28	14	3	31	6	1	36	6	0	32	16	4	34	8	2	32	11	0	5	12	14
J_3	29	13	2	33	8	0	39	3	0	33	11	2	34	11	0	31	8	0	8	9	10
Avg.	29.3	14.0	2.67	33.3	8.33	0.67	39.3	4.67	0	33.7	15.3	2.33	35.0	10.7	1	32.3	10.7	0	6.33	11.7	13.7
κ_{Fleiss}	0.73			0.68			0.74			0.71			0.72			0.76			0.73		

Table 3: Number of spammers detected in each bucket (B_1, B_2, B_3) by each judge (J_1, J_2, J_3) across each method. Last row reports the agreement of judges using Fleiss' multi-rater kappa (κ_{Fleiss}) for each method.

Profile evaluation of likely spammers via domain experts across three buckets. ASM variants ranks maximums # of spammers in B1 and almost 0 spammers in B3

Posterior Analysis

- Content Similarity (ψ^{CS}): Spammers have more content similarity.
- The expected value of content similarity (using cosine similarity) for non-spammers is 0.09, much lower than 0.7 for spammers

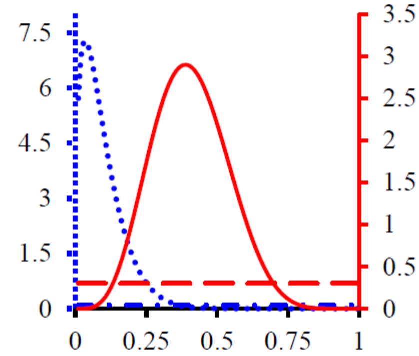


a) $\overline{\psi_S^{CS}} = 0.70$, $\overline{\psi_N^{CS}} = 0.09$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Max # Reviews (ψ^{MNR}): Spammers write more reviews/day.
- In expectation (using the MNR normalization constant of 21), spammers wrote close to $0.28 \times 21 \approx 5$ reviews/day while non-spammers wrote $0.11 \times 21 \approx 2$ reviews/day

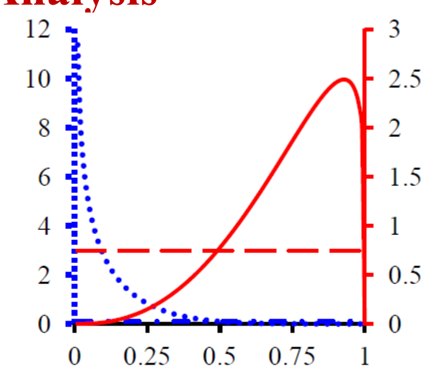


b) $\overline{\psi_S^{MNR}} = 0.28, \overline{\psi_N^{MNR}} = 0.11$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Burstiness (ψ^{BST}): Most non-spammers post reviews spanned over more than 28 days (Burstiness threshold).
- Expected account activity period for spammers is 0.75, i.e., $28 \times (1\text{-}0.75) \approx 7$ days after which account is not used

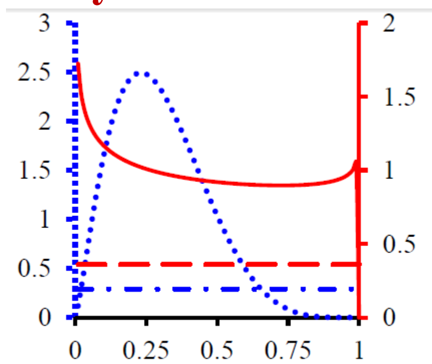


c) $\overline{\psi_S^{BST}} = 0.75, \overline{\psi_N^{BST}} = 0.10$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Ratio of First Reviews (ψ^{RFR}): separation between spammers and non-spammers is not so distinct in posting first reviews
- Means are close, although value for spammer population is higher

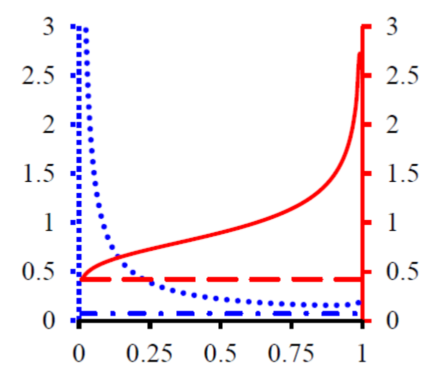


d) $\overline{\psi_S^{RFR}} = 0.36, \overline{\psi_N^{RFR}} = 0.29$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Duplicate Reviews (ψ^{DUP}): Spam reviews attain higher values (with density peak at extreme right) while non-spam reviews attain very low values (with peak density at extreme left)
- Duplicate reviews are sheer signs of spamming

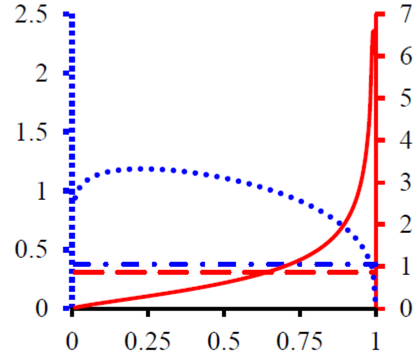


e) $\overline{\theta_S^{DUP}} = 0.42, \overline{\theta_N^{DUP}} = 0.07$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Extreme Rating (ψ^{EXT}): Spammers tend to give extreme ratings (1/5 stars)
- Non-spammers are rather evenly distributed as genuine reviewers usually have different rating levels

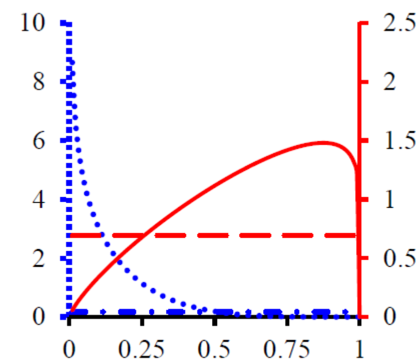


f) $\overline{\theta_S^{EXT}} = 0.86, \overline{\theta_N^{EXT}} = 0.36$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Posterior Analysis

- Rating Abuse (ψ^{RA}): Large percentage of spam reviews (70% in expectation) have been instances of imparting rating abuse (i.e., among the multiple reviews/ratings for the same product

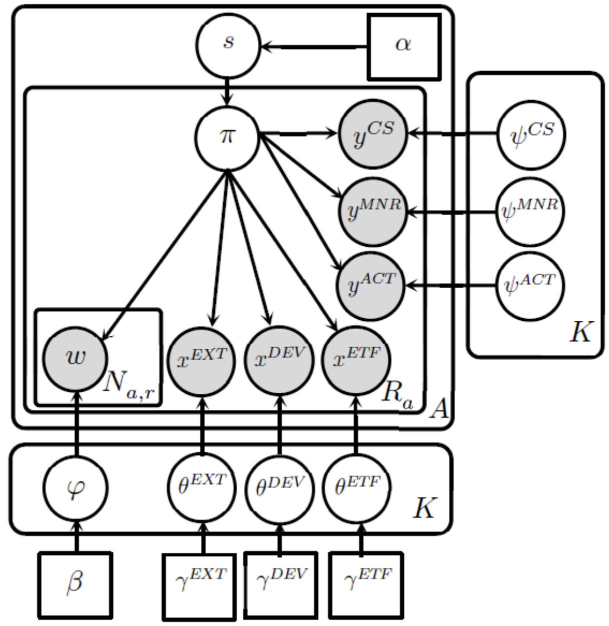


i) $\overline{\theta_S^{RA}} = 0.70, \overline{\theta_N^{RA}} = 0.13$

Figure 2: Density function (PDF) of estimated latent behvaior variables, $\psi^f \sim Beta$, $\theta^f \sim Beta$ corresponding to each author and review behavior feature, $f \in \{CS, MNR, BST, RFR\} \cup \{DUP, EXT, DEV, ETF, RA\}$. Estimated posterior densities for spam (in red/solid) and non-spam (in blue/dotted) are plotted with their respective scales (left:blue/dotted for non-spam and right: solid/red for spam). Also shown are the expected values for each latent behavior for spam (red/dashed) and non-spam (blue/dash-dot) in respective scales. Expected values are also reported in plot captions.

Generative Model Based Clustering on Reviews

- Unsupervised models for review spam detection [Mukherjee and Venkataraman, Tech. Rep. 2014].
- Model author and review spamicity from observed review behaviors.



Generative Model Based Clustering on Reviews

Algorithm	Feat.	E	P	Prec.	Rec.	F1	E	P	Prec.	Rec.	F1	E	P	Prec.	Rec.	F1
K-means (KM)	L	0.99	0.54	52.6	81.2	63.8	0.99	0.54	46.3	83.1	59.5	0.99	0.52	48.0	54.1	50.8
	B	–	–	–	–	–	0.99	0.52	47.6	85.0	61.0	0.99	0.52	48.1	54.2	50.9
	L+B	–	–	–	–	–	0.99	0.52	48.1	85.4	61.5	0.99	0.51	48.9	55.5	52.0
Single-Link HC	L	0.99	0.53	48.3	79.3	60.0	0.99	0.54	46.1	87.2	60.3	0.99	0.54	45.5	53.5	49.2
	B	–	–	–	–	–	0.99	0.54	46.3	88.0	60.6	0.99	0.55	45.9	54.0	49.6
	L+B	–	–	–	–	–	0.99	0.54	46.5	88.4	60.9	0.99	0.55	46.0	55.3	50.2
Complete-Link HC	L	0.99	0.51	49.6	83.1	62.1	0.99	0.52	48.1	85.4	61.5	0.99	0.52	47.5	54.6	50.8
	B	–	–	–	–	–	0.99	0.52	48.4	85.6	61.8	0.99	0.52	48.2	54.9	51.3
	L+B	–	–	–	–	–	0.99	0.52	49.1	85.9	62.5	0.99	0.52	48.6	55.2	51.7
LSM-UP	L+B	0.85	**0.70**	66.0	86.1	**74.6**	0.91	**0.63**	57.2	87.7	**69.2**	0.98	**0.56**	55.0	62.6	**58.4**
LSM-HE	L+B	0.83	**0.72**	66.1	89.0	**75.9**	0.83	**0.70**	63.7	89.2	**74.3**	0.97	**0.60**	59.2	64.1	**61.6**

(a) AMT Dataset (Ott et al., 2011) (b) Amazon (Mukherjee et al., 2012) (c) Yelp Restaurant Dataset

Table 2: Clustering performance comparison on various metrics: entropy (E), purity (P), and precision (Prec.), recall (Rec.), F1 on the fake (positive) class reported in % for the majority cluster. Metrics are reported for different clustering algorithms against different features (Feat.): (L)inguistics, (B)ehaviors. AMT data in Ott et al., (2011) does not have behavior information so values for B and L+B feature sets are nil. Improvements of LSM are significant (p<0.01, except entropy on the Yelp data which gives p<0.05) according to t-test over 50 runs.

Posterior on Fake/Non-Fake Language Models

Hotel Fake

Hotel Non Fake

Posterior on Fake/Non-Fake Language Models

Restaurant Fake

Restaurant Non Fake

Outline

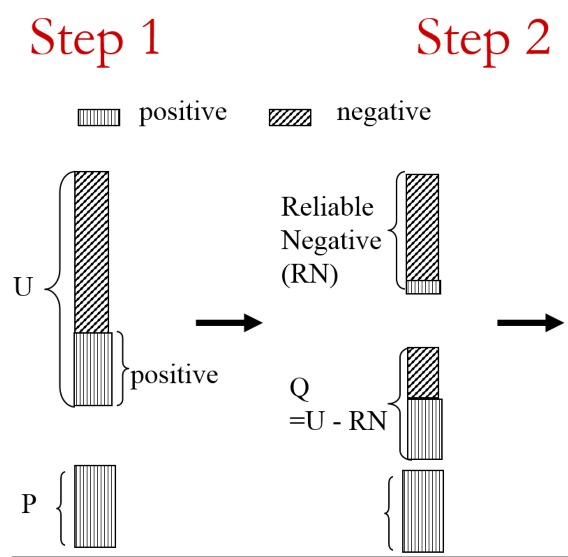
- ❑ Approach#2: Behavioral Modeling
- ❑ Approach#3: Statistical Modeling
  - **P.1: Latent Variable Models**
  - **P.2: Positive Unlabeled (PU) Learning**
  - **P.3: Collective PU Learning**
  - **P.4: Typed Markov Random Fields**
- ❑ …

PU Learning

- ❑ Positive examples: One has a set of examples of a class ***P***, and
- ❑ Unlabeled set: also has a set ***U*** of unlabeled (or mixed) examples with instances from *P* and also not from *P* (*negative examples*).
- ❑ Build a classifier: Build a classifier to classify the examples in *U* and/or future (test) data.
- ❑ Key feature of the problem: no labeled negative training data.
- ❑ This problem setting is often called as, PU-learning [Lee and Liu, ICML 2003]

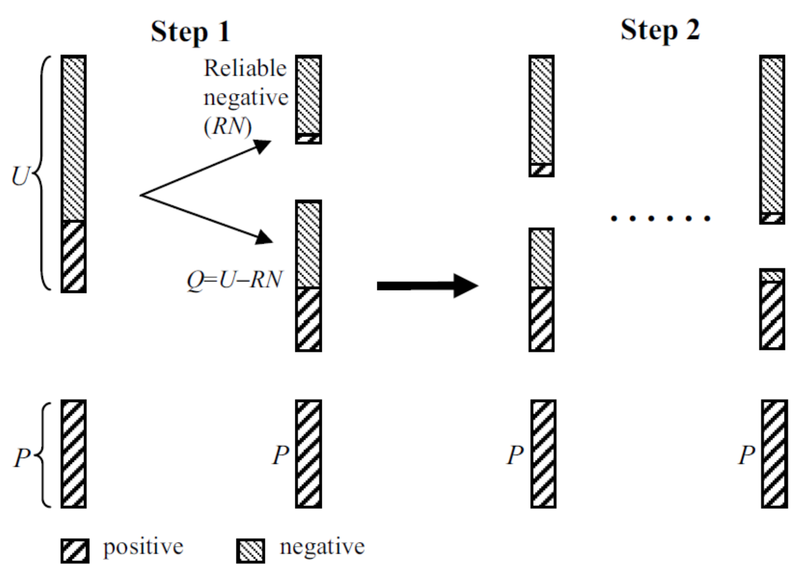
PU Learning

- Bootstrap Reliable Negatives from the unlabeled set, U (e.g., using **spy induction** by adding spies SP (positives into U) [Lee and Liu, 2003])
- Learn a new classifier using $P \setminus SP$, and $U \cup SP$
- Find most confident negative samples using a threshold
- This works because spies behave as their true label (i.e., positive class)



PU Learning

- Continue to refine U using multiple iterations



† Contains content originally appearing in [Liu, Web Data Mining 2008]

Deception Detection via PU Learning

- ❑ Large-scale gold-standard data for deceptive opinion spam is often limited, costly, time-consuming
- ❑ Q: How to leverage unlabeled data (reviews) to improve deception detection?
- ❑ PU Learning to the rescue

Using small scale positive (spam) labeled data, treat all unlabeled data containing both hidden positive (spam) and negative (non-spam) samples → Apply a PU Learning technique

PU Learning – Type I [Unlabeled as Negative]

- ❑ Treating Entire Unlabeled Data as Negative [Fusilier et al., ACL 2013]
- ❑ Experimented with on-class SVMs, and standard PU-Learning with NB and SVM as intermediate classifiers
- ❑ PU Learning outperformed one-class SVMs

One-class SVMs tend to perform better when there is very limited labeled data (~ 50 +ve samples) whereas PU-LEA works better when there are more +ve training smaples

PU Learning – Type II [Spy Induction]

- **Q: How to obtain reliable negative samples from the unlabeled data?**
- Add select positive examples as "spies" in the unlabeled set [Li et al., MICAI 2015]
- Learn a new classifier using P, RN and U

Spy Induction: "As spy examples are from P and are put into U as negatives in building the intermediate classifier, they (newly inserted spies) should behave similarly to the hidden positives in U. Hence, we can use them to find the reliable negative set RN from U

Extracting RN from U via Spy Induction

- Bootstrap RN
- Add spies
- Learn a new classifier using $P \backslash SP$, and $U \cup SP$
- Find most confident negative samples using a threshold
- This works because spies behave as their true lable (i.e., positive class)

```
RN ← ∅;
// Reliable negative set
SP ← Sample(P, s%);
// Spy set
Assign each example in P \ SP the class label +1;
Assign each example in U ∪ SP the class label −1;
C ← NB(P \ SP, U ∪ SP);
// Produce a NB classifier
Classify each u ∈ U ∪ SP using C;
Decide a probability threshold t using SP and l;
for each u ∈ U do
    if its probability Pr(+|u) < t then
        RN ← RN ∪ u
    end if
end for
```

EM via NB/SVM

- Bootstrap initial classifier using P and RN
- Iterate (until parameters stabilize):
  - E-step: obtain class likelihoods of unlabeled data
  - M-step: Maximize the likelihood of predicting the labels of the classifier in P, RN, and U
- Predict labels using the stable model parameters (estimated posterior, for NB)

Each document in P is assigned the class label $+1$;
Each document in RN is assigned the class label -1;
Learn an initial NB classifier f from P and RN;
do
    // E-Step
    for each document d_i in $U \setminus RN$ **do**
        Using the current classifier f to compute $Pr(c_j|d_i)$;
    end for
    // M-Step
    Learn a new NB classifier f from P, RN and $U \setminus RN$ using $Pr(c_j)$ and $Pr(w_t|c_j)$;
while the classifier parameters stablize
The last iteration of EM gives the final classifier f;
for each document d_i in U **do**
    if its probability $Pr(+|d_i) \geq 0.5$ **then**
        Output d_i as a positive document;
    else
        Output d_i as a negative document;
    end if
end for

Detection Performance on Chinese Fake Reviews

- Data courtesy of Dianping (Chinese Yelp)
- 3476 fake positive reviews, 3476 unknown (negative reviews)
- Feature set: unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Detection Performance on Chinese Fake Reviews

- Data courtesy of Dianping (Chinese Yelp)
- 3476 fake positive reviews, 3476 unknown (negative reviews)
- Feature set: unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Compared with the F score of 0.72 using bi-grams on Yelp restaurant reviews in [Mukherjee et al., ICWSM 2013], the detection performance is lower.

Q: What is the reason?

Detection Performance on Chinese Fake Reviews

- Data courtesy of Dianping (Chinese Yelp)
- 3476 fake positive reviews, 3476 unknown (negative reviews)
- Feature set: unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

(1) Dianping reviews are much shorter than Yelp reviews and thus have less information for learners.

(2) Chinese words are not naturally separated by white spaces. Errors produced by word segmentation would lead to poorer linguistic features.

Detection Performance using PU-Learning

- PU-LEA [Fusilier et al., ACL 2013] (Using all U as negative set)
- Spy induction – [Li et al., MICAI 2014]
- Feature set: Unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Key Observations:

(1) Spy induction outperforms PU-LEA at 98% confidence (p<0.02).

Detection Performance using PU-Learning

- PU-LEA [Fusilier et al., ACL 2013] (Using all U as negative set)
- Spy induction – [Li et al., MICAI 2014]
- Feature set: Unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Key Observations:

(1) Spy induction outperforms PU-LEA at 98% confidence (p<0.02).

(2)Spy-EM (with NB as intermediate classifier) outperforms in Recall.

Detection Performance using PU-Learning

- PU-LEA [Fusilier et al., ACL 2013] (Using all U as negative set)
- Spy induction – [Li et al., MICAI 2014]
- Feature set: Unigrams and bigrams

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Key Observations:

(1) Spy induction outperforms PU-LEA at 98% confidence (p<0.02).

(2)Spy-EM (with NB as intermediate classifier) outperforms in Recall.

(3) Spy-SVM (with SVM as intermediate classifier) outperforms in F1

Behavioral Analysis of False Positives

- PU-Learning yields significantly higher recall than SVM, but lower precision.
- **Q: Is low precision is caused by hidden fake reviews in the unlabeled set?**

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Behavioral Analysis of False Positives

- PU-Learning yields significantly higher recall than SVM, but lower precision.
- **Q: Is low precision is caused by hidden fake reviews in the unlabeled set?**

Table 1. 5-fold CV results

	SVM			PU-LEA			Spy+EM			Spy+SVM		
	P	R	F	P	R	F	P	R	F	P	R	F
Unigrams	**0.54**	0.51	0.52	**0.54**	0.53	0.54	0.44	**0.86**	0.58	0.49	0.77	**0.60**
Bigrams	0.54	0.52	0.52	**0.55**	0.54	0.55	0.44	**0.89**	0.59	0.53	0.72	**0.61**

Does transferring some False Positives (FP) to True Positive (TP) (because those reviews are indeed deceptive as attested by other behavioral signals) increase the precision?

How to decide which FP reviews to transfer?

Behavioral Analysis of False Positives

- Two behavioral heurists of spamming:
  - Max content similarity (MCS).
  - Average # reviews/day (ANR)

Table 3. Label adjustments by moving false positive (FP) to true positive (TP). MCS ≥ 0.8 is used for all experiments

	SVM			PU-LEA			LPU (Spy+EM)			LPU (Spy+SVM)		
ANR	#FP1	#FP2	#MV	#FP1	#FP2	#MV	#FP1	#FP2	#MV	#FP1	#FP2	#MV
≥ 2	49	0	49	41	0	41	170	228	295	86	114	149
≥ 3	49	0	49	41	0	41	170	110	227	86	56	115
≥ 4	49	0	49	41	0	41	170	62	201	86	31	101
≥ 5	49	0	49	41	0	41	170	43	192	86	22	97
≥ 6	49	0	49	41	0	41	170	34	185	86	17	94

Set MCS > 0.8 and vary the threshold for ANR

#FP1: reviews meeting MCS criteria

#FP2: Reviews meeting ANR criteria

#MV: Reviews satififsying either one of the criteria

Performance Gains upon Transferring FP → TP

- Significant gains in precision and F1 of Spy+EM and Spy+SVM
- Improvements of SVM and PU-LEA are smaller than Spy models

Table 4. Results using bigrams after moving false positive (FP) to true positive (TP). MCS ≥ 0.8 is used for all experiments

	SVM			PU-LEA			Spy+EM			Spy+SVM		
ANR	P	R	F	P	R	F	P	R	F	P	R	F
≥ 2	0.63	0.52	0.57	0.62	0.54	0.58	0.59	**0.89**	**0.71**	**0.68**	0.72	0.70
≥ 3	0.63	0.52	0.57	0.62	0.54	0.58	0.55	**0.89**	0.68	**0.64**	0.72	**0.68**
≥ 4	**0.63**	0.52	0.57	0.62	0.54	0.58	0.53	**0.89**	0.66	**0.63**	0.72	**0.67**
≥ 5	**0.63**	0.52	0.57	0.62	0.54	0.58	0.52	**0.89**	0.66	0.62	0.72	**0.67**
≥ 6	**0.63**	0.52	0.57	0.62	0.54	0.58	0.52	**0.89**	0.65	0.62	0.72	**0.67**

Inference - Spy induction can discover hidden positives (fakes) in unlabeled data.

Performance Gains upon Transferring FP → TP

- Significant gains in precision and F1 of Spy+EM and Spy+SVM
- Improvements of SVM and PU-LEA are smaller than Spy models

Table 4. Results using bigrams after moving false positive (FP) to true positive (TP). MCS ≥ 0.8 is used for all experiments

	SVM			PU-LEA			Spy+EM			Spy+SVM		
ANR	P	R	F	P	R	F	P	R	F	P	R	F
≥ 2	0.63	0.52	0.57	0.62	0.54	0.58	0.59	**0.89**	**0.71**	**0.68**	0.72	0.70
≥ 3	0.63	0.52	0.57	0.62	0.54	0.58	0.55	**0.89**	0.68	**0.64**	0.72	**0.68**
≥ 4	**0.63**	0.52	0.57	0.62	0.54	0.58	0.53	**0.89**	0.66	**0.63**	0.72	**0.67**
≥ 5	**0.63**	0.52	0.57	0.62	0.54	0.58	0.52	**0.89**	0.66	0.62	0.72	**0.67**
≥ 6	**0.63**	0.52	0.57	0.62	0.54	0.58	0.52	**0.89**	0.65	0.62	0.72	**0.67**

Dianping's Fraud Detection Team agreed that those moved FP to TP are indeed true positive (spam) that their classifier could not catch!

Beyond PU Learning

- Drawbacks of PU Learning:
- Flat – Static Data (as opposed to Linked/Graph based Data)
- Premature Convergence – converges too early before enough hidden positives are discovered if the positives are not very close to the hidden positives in the unlabeled data

Fake reviews might share IP addresses (latent sockpuppet) in the embedded network structure.

How to leverage PU learning with network information?

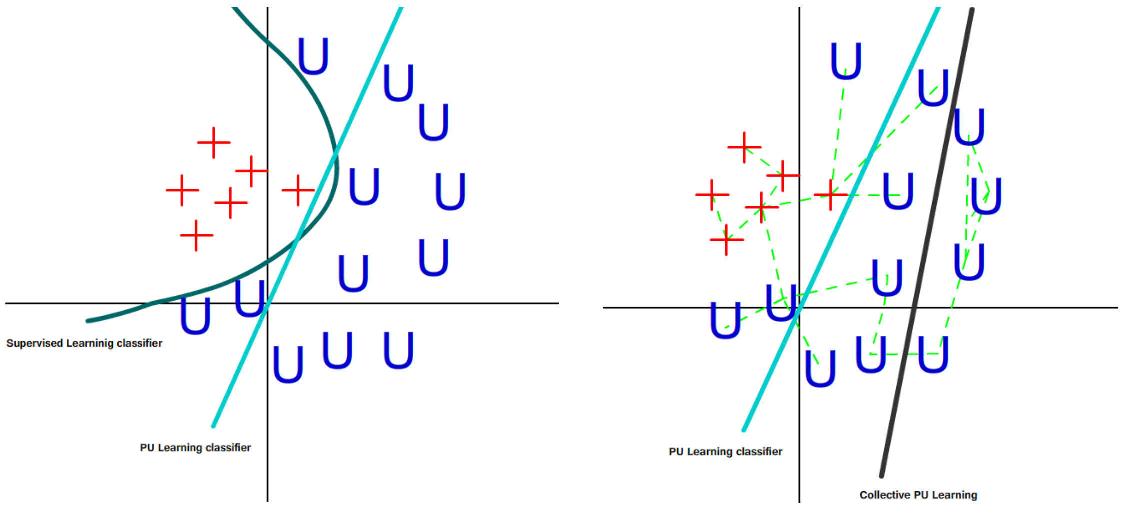
Outline

- Approach#2: Behavioral Modeling
- Approach#3: Statistical Modeling
  - **P.1: Latent Variable Models**
  - **P.2: Positive Unlabeled (PU) Learning**
  - **P.3: Collective PU Learning**
  - **P.4: Typed Markov Random Fields**
- …

Collective Positive Unlabeled Learning

Supervised Learning and PU Learning

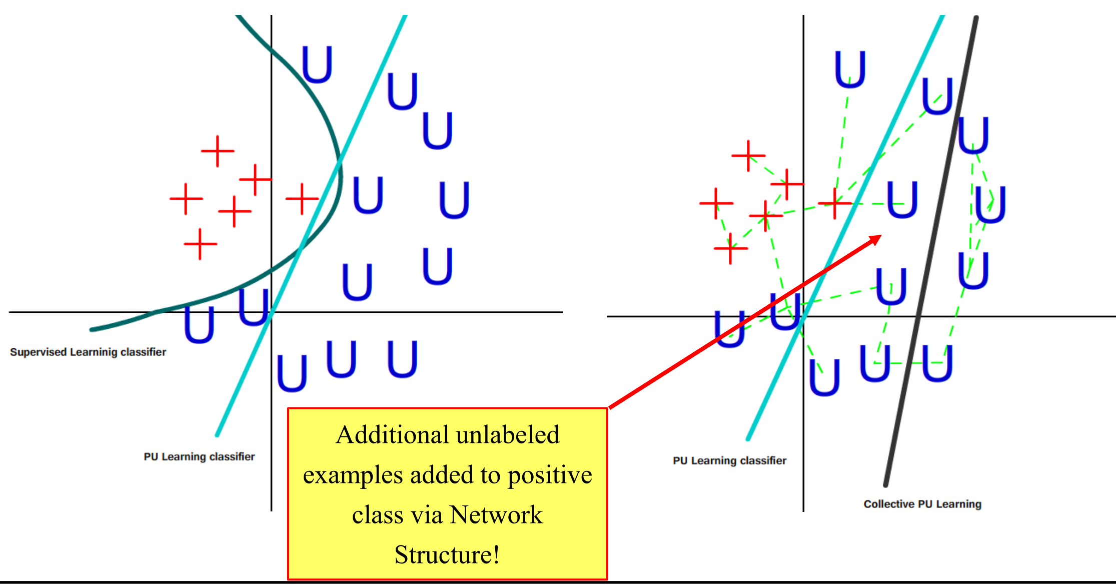
PU Learning and Collective PU Learning



Collective Positive Unlabeled Learning

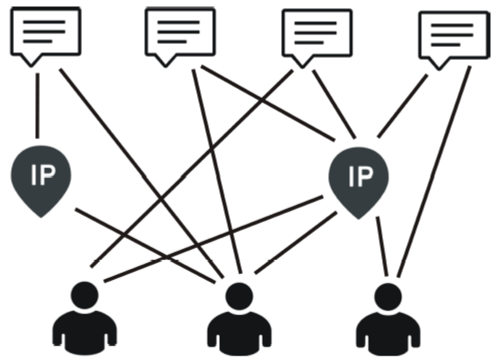
Supervised Learning and PU Learning

PU Learning and Collective PU Learning



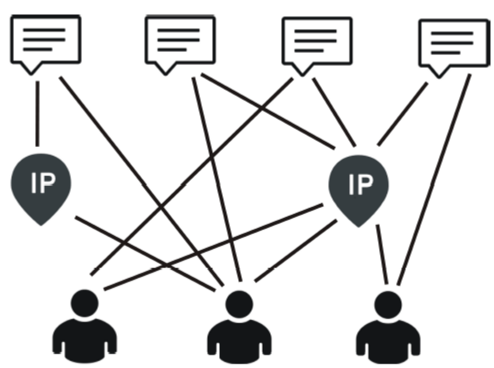
Opinion Spam Detection via Collective PU Learning

- Using IP addresses as bridges for users and reviews [Li et al., ICDM 2014]:
- Heterogeneous Network of Users, IP, Reviews
- one review only belongs to one user and one IP address, but users and IPs can connect to more than one entities of other types.



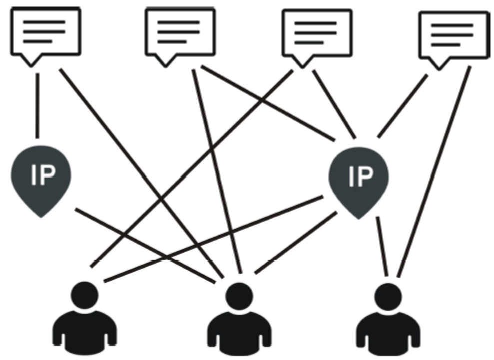
Collective Classification

- Collective classifiers (CC) [Sen et al., Tech Rep. 2008] serve the baseline framework
- Conventional classifiers (CC) on graph nodes only use the local features of that node
- CC such as ICA [Sen et al., Tech Rep. 2008] trains a local classifier leveraging the observed local (node) features and estimated labels of its neighbors.



Collective Classification

- Unlabeled data (user-IP-reviews) treated as negative set.

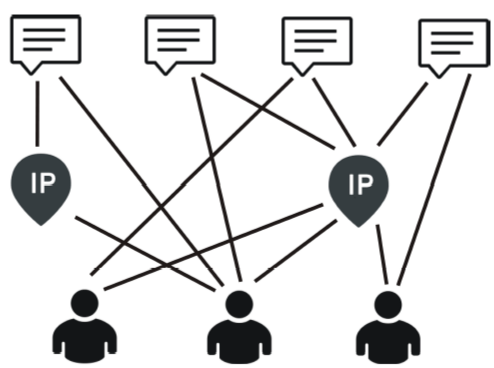


Reviews sharing same IP/user likely to have similar labels

Multi-Type Heterogeneous Collective Classification

- Multi-Type Heterogeneous Collective Classification (MHCC) on user-IP-review network [Li et al., ICDM 2014] serve as the baseline

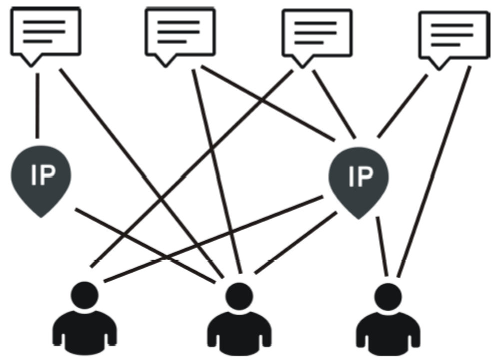
- Conventional classifiers on graph nodes only use the local features of that node



Reviews sharing same IP/user likely to have similar labels

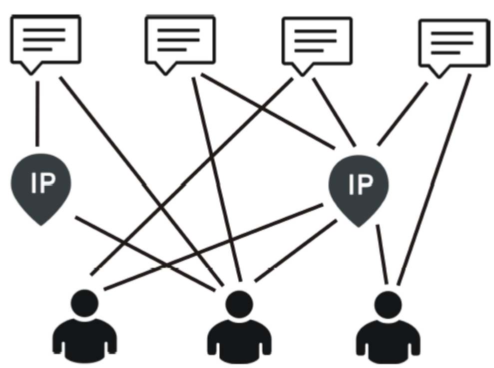
Multi-Type Heterogeneous Collective Classification

- Step 1: Bootstrap review classifier. Estimate user and IP labels from majority class label of their related reviews.
- Step 2: During iterative prediction, construct a relational feature matrix M from the estimate labels of the neighboring nodes.
- Step 3 Then train three different relational (local) classifiers for reviews, users and IPs



Feature Set for MHCC

- Linguistic:
  - Unigram
  - Bigram
- Behavioral:
  - Maximum Number of Reviews per day
  - Total Number of Reviews
  - Number of active days
  - Average number of reviews per active day
  - Percentage of Positive review
  - Average rating
  - Rating deviation
  - Average length of reviews
  - Maximum content similarity

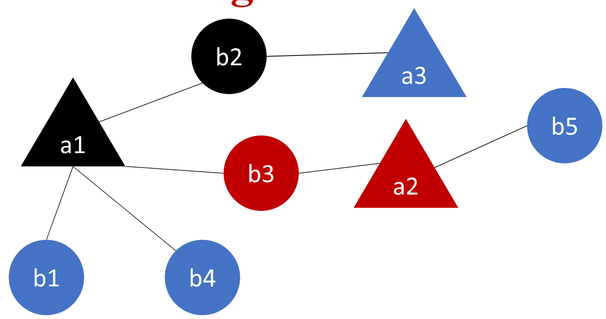


Collective PU Learning

Labels using bootstrap classifier

	f_1	f_2	...	f_a	b_1	b_2	b_3	b_4	b_5	y
a_1					-1	-1	1	-1	0	-1
a_2		A			0	0	1	0	-1	1
a_3					0	-1	0	0	0	?

	f_1	f_2	...	f_b	a_1	a_2	a_3	y
b_1					-1	0	0	?
b_2					-1	0	-1	-1
b_3		B			-1	1	0	1
b_4					-1	0	0	?
b_5					0	1	0	?



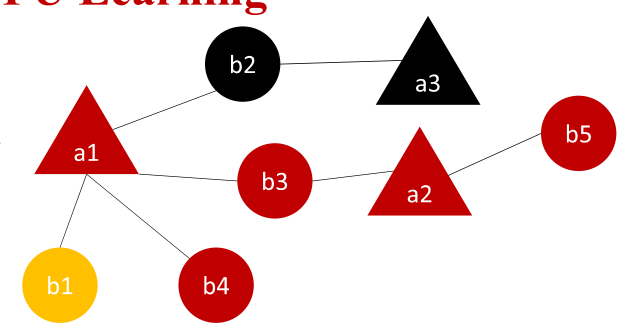
Shape : Entity Type
Color : Class
unknown (blue) [to estimate]
positive class (red) [hard label]
negative class (black) [soft label]

Collective PU Learning

Label change using CC Pr(b1=+1) is low but Pr(b4=+1) is high → So use b4 for training

	f_1	f_2	...	f_a	b_1	b_2	b_3	b_4	b_5	y
a_1					-1	-1	1	-1	0	1
a_2		A			0	0	1	0	-1	1
a_3					0	-1	0	0	0	?

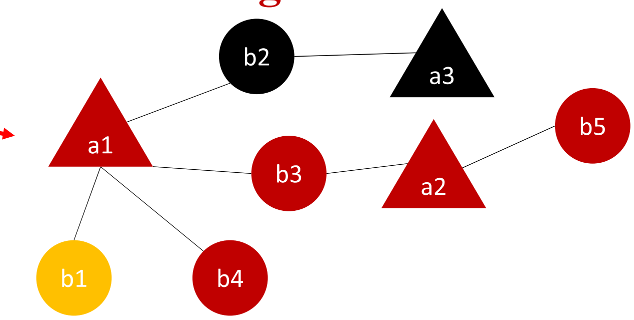
	f_1	f_2	...	f_b	a_1	a_2	a_3	y
b_1					1	0	0	?
b_2					1	0	-1	-1
b_3		B			1	1	0	1
b_4					1	0	0	?
b_5					0	1	0	?



Shape : Entity Type
Color : Class
unknown (blue) [to estimate]
positive class (red) [hard label]
negative class (black) [soft label]

Collective PU Learning

Iterate, increase labels, iterate, until classifier parameters stabilize



	f_1	f_2	...	f_a
a_1	A			
a_2				
a_3				

b_1	b_2	b_3	b_4	b_5
-1	-1	1	-1	0
0	0	1	0	-1
0	-1	0	0	0

y
1
1
?

	f_1	f_2	...	f_b
b_1	B			
b_2				
b_3				
b_4				
b_5				

a_1	a_2	a_3
1	0	0
1	0	-1
1	1	0
1	0	0
0	1	0

y
?
-1
1
?
?

Shape : Entity Type
Color : Class
unknown (blue) [to estimate]
positive class (red) [hard label]
negative class (black) [soft label]

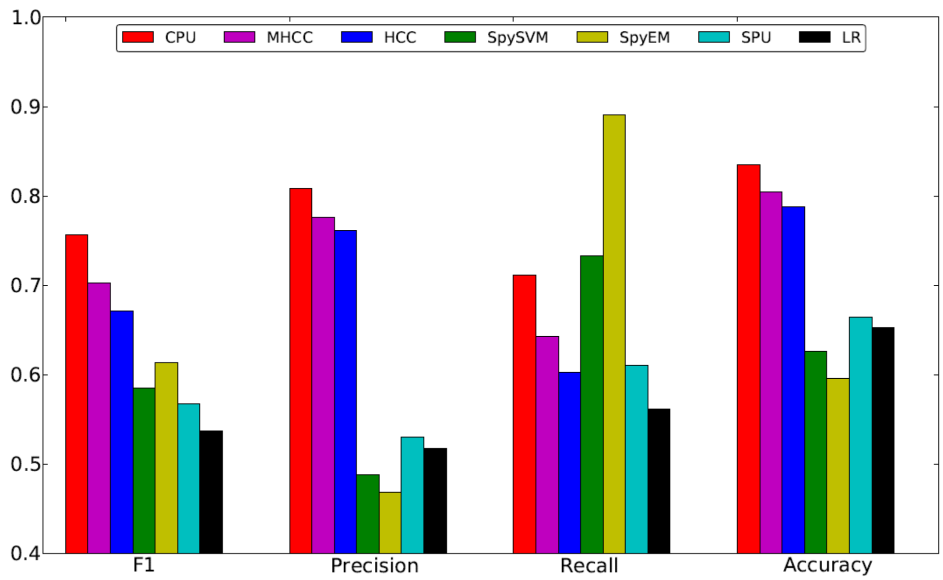
Shanghai Restaurant Dataset

- Dianping's Data Statistics
- 500 restaurants in Shanghai between November 1st, 2011 and November 28th, 2013

	Fake reviews	Unlabeled reviews	Total
No. of reviews	3523	6242	9765
No. of unique users	3310	5894	9067
No. of unique IPs	1314	4564	5535
No. of reviews per user	1.064	1.059	1.077
No. of reviews per IP	2.681	1.368	1.764
Avg No. of Chinese Characters	75.60	91.10	85.50
Avg No. of Chinese Words	53.17	63.21	59.59

Performance Analysis of Collective PU Learning

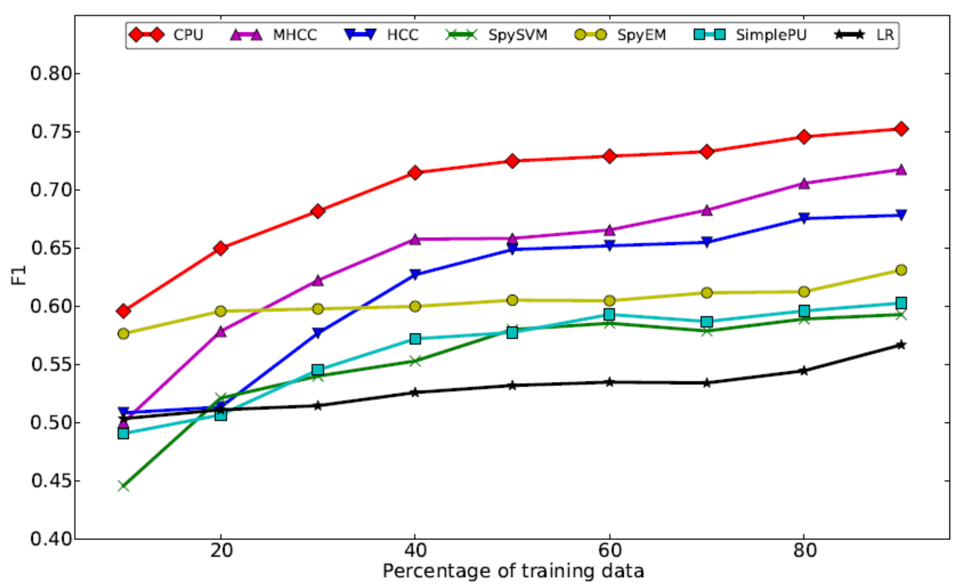
- Result due to [Li et al., ICDM 2014]
- Spy-SVM, Spy-EM overshot the positives.
- CPU outperforms in F1 and Acc



† Contains content originally appearing in [Li et al., ICDM 2014]

Sensitivity of Training Set

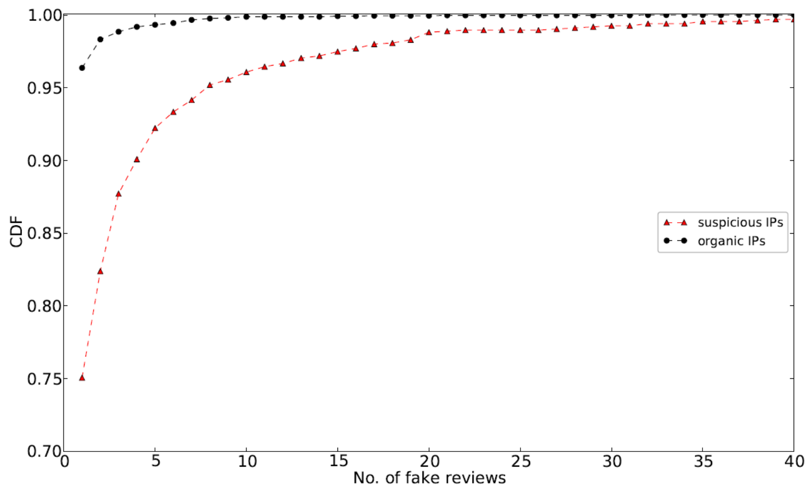
- Result due to [Li et al., ICDM 2014]
- F1 vs. different training ratios (% of pos training sample)
- CPU retains its lead



† Contains content originally appearing in [Li et al., ICDM 2014]

IP distribution for fakes

- ❑ CDF of the number of fake reviews for suspicious IPs v.s. organic IPs
- ❑ Majority (97%) of organic IPs bounded by very few fakes
- ❑ Only 75% of suspicious IPs bounded by few fakes



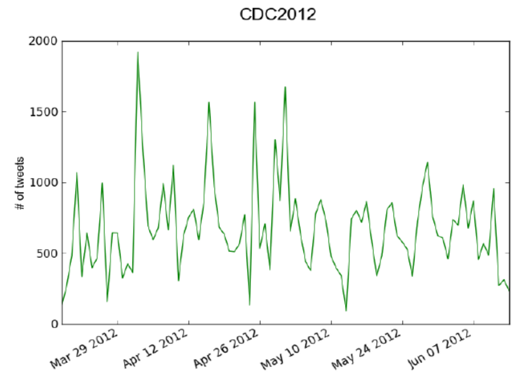
Several fake reviews appears form a pool of suspicious IPs → exploiting network effects is useful

Outline

- ❑ Approach#2: Behavioral Modeling
- ❑ Approach#3: Statistical Modeling
  - **P.1: Latent Variable Models**
  - **P.2: Positive Unlabeled (PU) Learning**
  - **P.3: Collective PU Learning**
  - **P.4: Typed Markov Random Fields**
- ❑ …

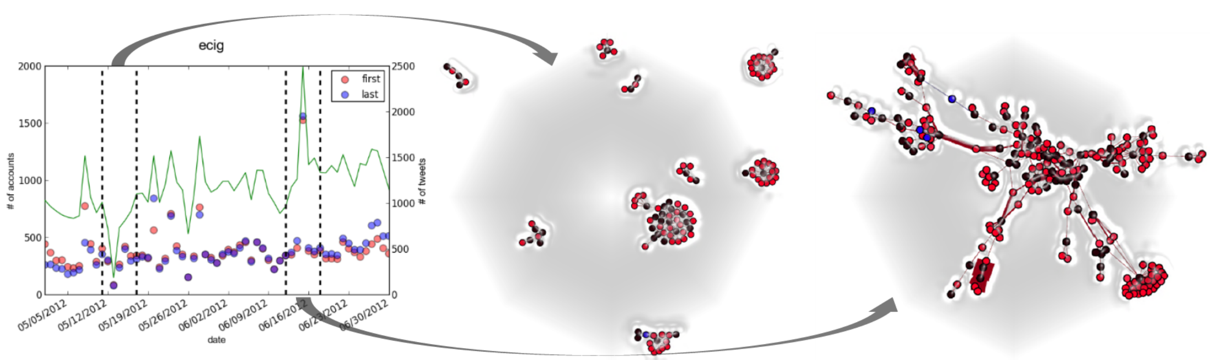
Opinion Spamming Campaigns: E-cig marketing on Twitter

- Spammers promote opinons/urls in twitter to boost sales
- These promoted content often results in retweet bursts containing the entity url being promoted
- Campaings can be organic (CDC promoting no smoking) vs. marketing campaigns of e-cig
- Campaings invariably have bursts



Network Structures of Opinion Spamming Campaigns

- # users with first post in that day (red dots) correlate with # users with last post in that day (blue dots)
- This further correlates with # of tweets on the entity

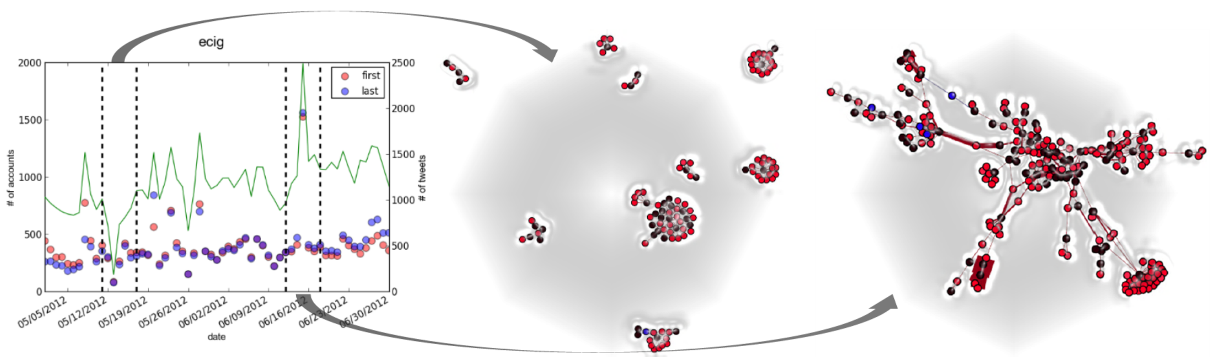


(a) Campaign Timeline (b) Campaign Network Structure at Initiation (c) Campaign Network Structure at Maturity

Network Structures of Opinion Spamming Campaigns

- # users with first post in that day (red dots) correlate with # users with last post in that day (blue dots)
- This further correlates with # of tweets on the entity

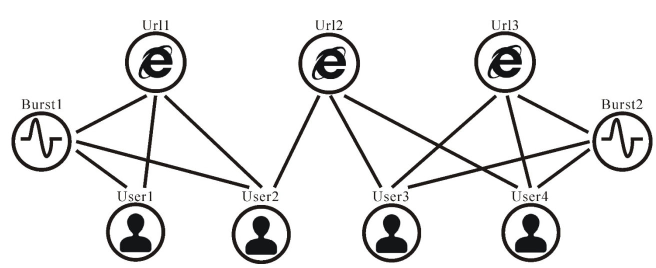
Network structure shows spamming campaigns that start of as individual small groups to form larges spam communities



(a) Campaign Timeline (b) Campaign Network Structure at Initiation (c) Campaign Network Structure at Maturity

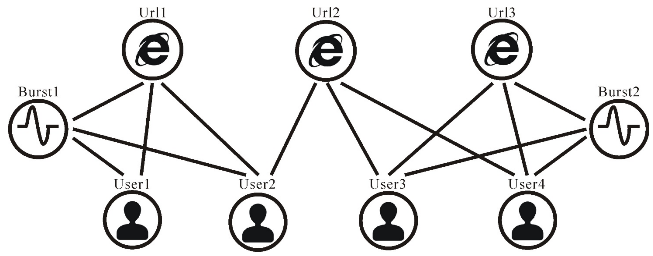
Heterogeneous Network of Opinion Spamming Campaigns

- A user is either a promoter or a non-promoter.
- A URL is either a promoted or organic URL. (URL shorten → expanded/full URL) ~50% contain URLs
- A burst is either a planned or normal burst. (sudden popularity by normal users, deliberated pushing by promoters, triggered by external event)



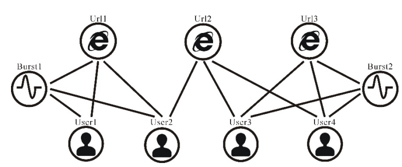
Modeling Opinion Spamming Campaigns via MRFs

- Campaign detection via Typed-MRFs [Li et al., ICDM 2014]
- MRF → Typed MRFs. State spaces of node types are:
- A user is either a promoter or a non-promoter.
- A URL is either a promoted or organic URL.
- A burst is either a planned or normal burst.



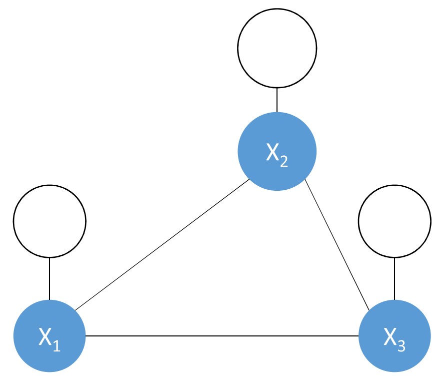
Modeling Opinion Spamming Campaigns via MRFs

Symbol	Definition
V	Set of nodes in the graph
E	Set of edges in the graph
T	Mapping from nodes to node types
H	Set of types of nodes
v_i	i-th node or random variable in the graph
t_i	Type of node i, $t_i \in H$
S_{t_i}	Set of states node i can be in
$\psi_i(\sigma_i \mid t_i)$	Prior of node i in state σ_i
$\psi_{i,j}(\sigma_i, \sigma_j \mid t_i, t_j)$	Edge potentials for node i of type t_i in state σ_i and node j of type t_j in σ_j
$m_{i \to j}(\sigma_j \mid t_j)$	Message from node i to node j expressing node i's belief to node j being in state σ_j
$b_i(\sigma_i \mid t_i)$	Belief of node i in state σ_i



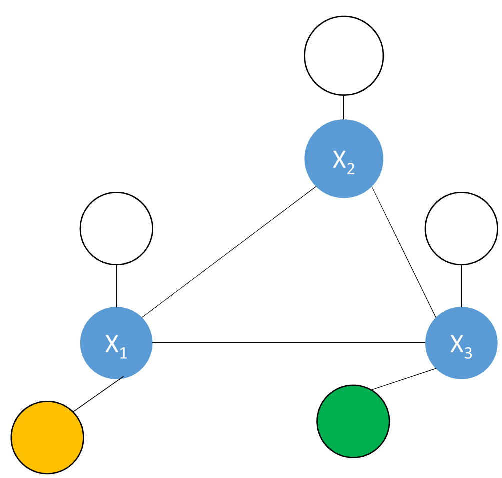
MRF Recap

- ❑ Markov Random Fields (also called Markov Networks) is an undirected graphical model that deals with inference problems with uncertainty in observed data.
- ❑ Nodes : random variable, edges : statistical relations.
- ❑ A set of potential functions are defined on the cliques of the graph to measure the compatibility among the nodes..



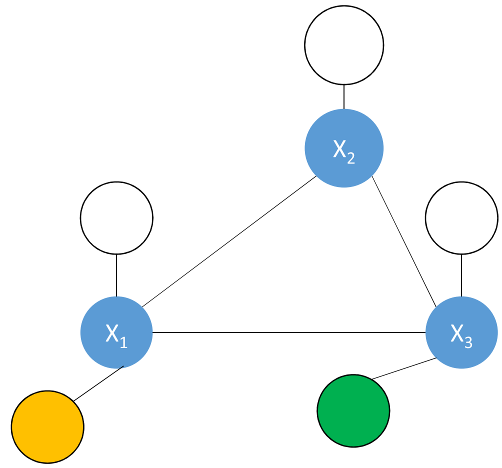
MRF Recap

- ❑ In our problem, we use the pairwise MRF in which the potential functions are defined on each of the edges.
- ❑ The label of node depends on the local features (yellow nodes) as well as its neighbors (green nodes).



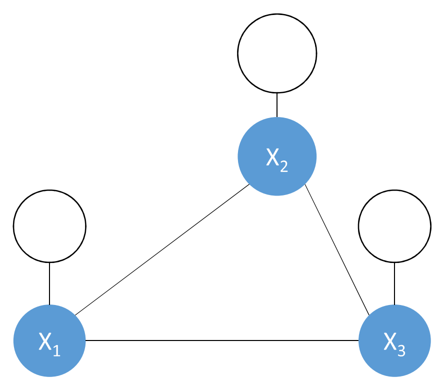
MRF Recap

- The label of node depends on the local features (yellow nodes) as well as its neighbors (green nodes). Example in smoking domain:
- Local features: color of teeth, height, weight, health condition, age, income, gender and so on.
- Compatibility: People who smoke tend to have friends who are smokers too.



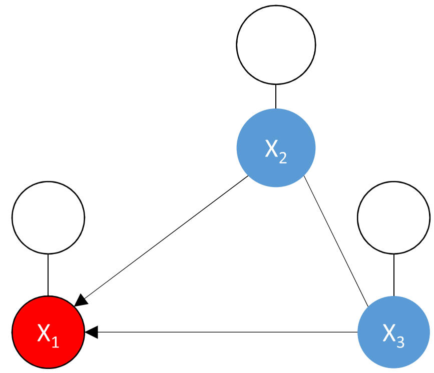
Inference via Loopy Belief Propagation

- LBP is generally used to estimate the marginal probability distribution of each node given the prior knowledge and potential functions
- Message $m_{ij}(x_j)$, node x_i 's belief of the node x_j's state



Inference via Loopy Belief Propagation

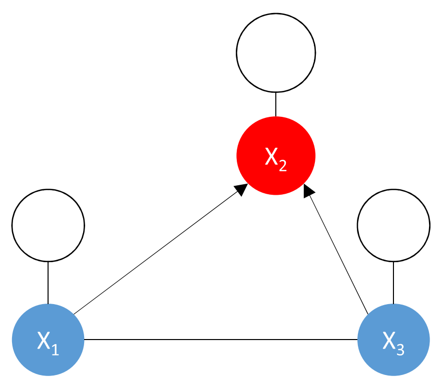
- Message passing – each node updates its belief of the state of all other nodes via passing messages of the form
- Message $m_{ij}(x_j)$, node x_i 's belief of the node x_j's state
- z_1 normalization factor to ensure $\sum_j m_{ij}(x_j) = 1$



$$m_{i\to j}(\sigma_j) = z_1 \sum_{\sigma_i \in S} \psi_{i,j}(\sigma_i, \sigma_j)\psi_i(\sigma_i) \prod_{k \in N(i)\setminus j} m_{k\to i}(\sigma_i)$$

Inference via Loopy Belief Propagation

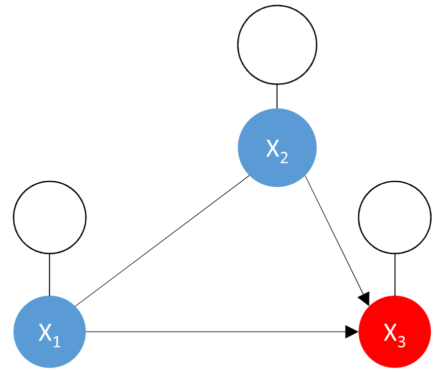
- Iterate via message passing
- Message $m_{ij}(x_j)$, node x_i 's belief of the node x_j's state
- z_1 normalization factor to ensure $\sum_j m_{ij}(x_j) = 1$



$$m_{i\to j}(\sigma_j) = z_1 \sum_{\sigma_i \in S} \psi_{i,j}(\sigma_i, \sigma_j)\psi_i(\sigma_i) \prod_{k \in N(i)\setminus j} m_{k\to i}(\sigma_i)$$

Inference via Loopy Belief Propagation

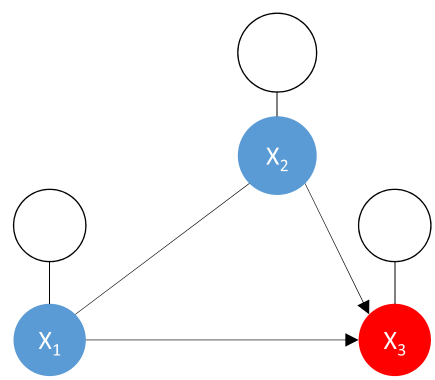
- ❑ Parallel updates are possible for speedup
- ❑ Iterate until convergence



$$m_{i\to j}(\sigma_j) = z_1 \sum_{\sigma_i \in S} \psi_{i,j}(\sigma_i,\sigma_j)\psi_i(\sigma_i) \prod_{k\in N(i)\backslash j} m_{k\to i}(\sigma_i)$$

Inference via Loopy Belief Propagation

- ❑ Belief read out upon convergence
- ❑ "Pool" beliefs of all neighboring nodes to arrive at the final posterior belief on the state of the current state
- ❑ z_2 normalization factor



$$b_i(\sigma_i) = z_2\ \psi_i(\sigma_i) \prod_{k\in N(i)} m_{k\to i}(\sigma_i)$$

Node Potentials (Priors)

- User type node priors derived using local discriminative classifiers (e.g., LR)
- Features:
  - number of URLs per tweet
  - number of hashtags per tweet
  - number of user mentions per tweet
  - percentage of retweets
  - maximum/minimum/average number of tweets per day
  - maximum/minimum/average time interval between two consecutive tweets

$$P_{user}(+) = \frac{1}{1 + e^{-\beta_0 - \sum_{j=1}^{k} \beta_j x_j}}$$

$$P_{user}(-) = \frac{e^{-\beta_0 - \sum_{j=1}^{k} \beta_j x_j}}{1 + e^{-\beta_0 - \sum_{j=1}^{k} \beta_j x_j}}$$

Node Potentials (Priors)

- Url/burst node type node priors derived using estimated count variables
- n^+: # of estimated promoters
- n^-: # of estimated organic users

$$P_{url}(+) = \frac{n^+ + \alpha}{n^+ + n^- + 2\alpha}$$

$$P_{url}(-) = \frac{n^- + \alpha}{n^+ + n^- + 2\alpha}$$

Edge Potentials (Message Factors)

- Url/Burst edge potentials
- A user-burst edges denote user posting tweets in the burst.
- Planned bursts contain primarily promoters
- Normal bursts are mostly formed by normal users who are attracted by the campaign.

	t_j = Burst	
t_i = URL	planned	normal
promoted	$0.5 + \epsilon$	$0.5 - \epsilon$
organic	$0.5 - \epsilon$	$0.5 + \epsilon$

Edge Potentials (Message Factors)

- Usr/Url edge potentials
- User-URL edge implies the user has tweeted the URL at least once.
- Heavily promoted URL → user likely to be a promoter
- Non promoted URL → user likely to be a non-promoter

	t_j = URL	
t_i = User	promoted	organic
promoter	$1 - 2\epsilon$	2ϵ
non-promoter	2ϵ	$1 - 2\epsilon$

Edge Potentials (Message Factors)

- Usr/Burst edge potentials
- URL-burst edge indicates the URL has been tweeted at least once in the burst
- URLs mentioned within a planned burst are likely to be promoted
- URLs in a normal burst are likely to be organic

	t_j = Burst	
t_i = User	planned	normal
promoter	$0.5 + \epsilon$	$0.5 - \epsilon$
non-promoter	$0.5 - \epsilon$	$0.5 + \epsilon$

Edge Potentials (Message Factors)

- Usr/Usr edge potentials
- **Q: How to connect user with other users (latent sockpuppets)?**

Edge Potentials (Message Factors)

- Usr/Usr edge potentials
- **Q: How to connect user with other users (latent sockpuppets)?**
- Define:
  - Text similarity
  - URL mention similarity
  - Following similarity
- Model user similarity by average of all similarities

$$CS_{i,j} = cosine(avg(tweets_i), avg(tweets_j))$$

$$US_{i,j} = \frac{|r_i \cap r_j|}{|r_i \cup r_j|}$$

$$FS_{i,j} = \frac{|f_i \cap f_j|}{|f_i \cup f_j|}$$

Edge Potentials (Message Factors)

- Usr/Usr edge potentials
- **Q: How to connect user with other users (latent sockpuppets)?**
- Define:
  - Text similarity
  - URL mention similarity
  - Following similarity
- Model user similarity by average of all similarities

Links are added between two users when avg. similarity of users is higher than a threshold. Intuitively, if a user is connected with a promoter, then he/she is also likely to be a promoter.

Edge Potentials (Message Factors)

- Usr/Usr edge potentials

	t_j = User	
t_i = User	promoter	non-promoter
promoter	$0.5 + \epsilon$	$0.5 - \epsilon$
non-promoter	0.5	0.5

- **Q: How to connect user with other users (latent sockpuppets)?**

- Define:
  - Text similarity
  - URL mention similarity
  - Following similarity

- Model user similarity by average of all similarities

Links are added between two users when avg. similarity of users is higher than a threshold. Intuitively, if a user is connected with a promoter, then he/she is also likely to be a promoter.

Overall Algorithm

- Typed message passing:

$$m_{i\to j}(\sigma_j|t_j) = z_1 \sum_{\sigma_i \in S} \psi_{i,j}(\sigma_i, \sigma_j|t_i, t_j)\psi_i(\sigma_i|t_i) \prod_{k\in N(i)\backslash j} m_{k\to i}(\sigma_i|t_i)$$

- Belief read out:

$$b_i(\sigma_i|t_i) = z_2\ \psi_i(\sigma_i|t_i) \prod_{k\in N(i)} m_{k\to i}(\sigma_i|t_i)$$

Algorithm 1 **The overall algorithm**

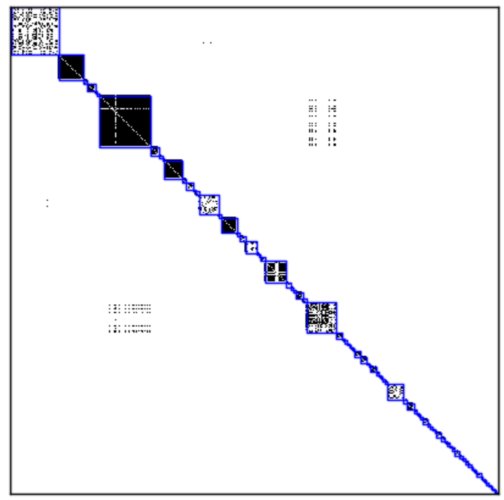
Input: A set of labeled users U_{train} for training
A set of tweets D on a particular topic
The propagation matrices $\psi_{i,j}(\sigma_i, \sigma_j|t_i, t_j)$
Output: Probability estimate of every user being a promoter

```
Train a classifier c from D and U_train
Apply c on all the unlabeled users to obtain the user priors
  (node potentials): ψ_i(σ_i|t_i = user)
Calculate URL and burst priors ψ_i(σ_i|t_i = URL) and
  ψ_i(σ_i|t_i = burst) using Eqn. 10 and 11.
Build the User-URL-Burst graph G(V, T, E) from D
for (v_i, v_j) ∈ E do
    for all states σ_j of v_j do
        m_{i→j}(σ_j|t_j) ← 1
    end for
end for
while not converged do
    for (v_i, v_j) ∈ E do
        for all states σ_j of v_j do
            update m_{i→j}(σ_j|t_j) in parallel using Eqn. 3.
        end for
    end for
end while
Calculate the final belief of every node in all states
  b_i(σ_i|t_i) using Eqn. 4.
Output the probability of every user being a promoter
  b_i(σ_i = promoter|t_i = user).
```

User-User Similarity

- ❑ Similarity distribution for $sim > 0.9$
- ❑ E-cig (spam) campaign data has several dense cliques → presence of twitter bots, sockpuppets

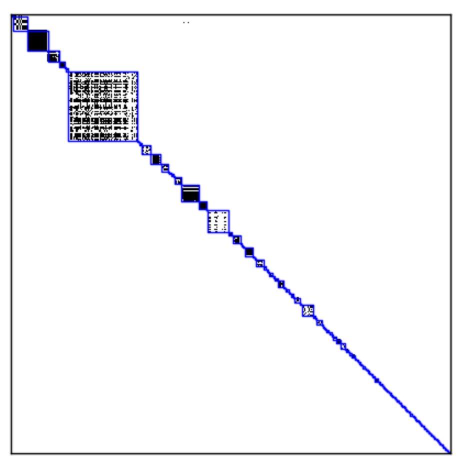
Dataset E-cig Campaign



User-User Similarity

- ❑ Similarity distribution for $sim > 0.9$
- ❑ Large block of similar users show affiliations of CDC with other health research institutes

Dataset CDC Stop Smoking Campaign



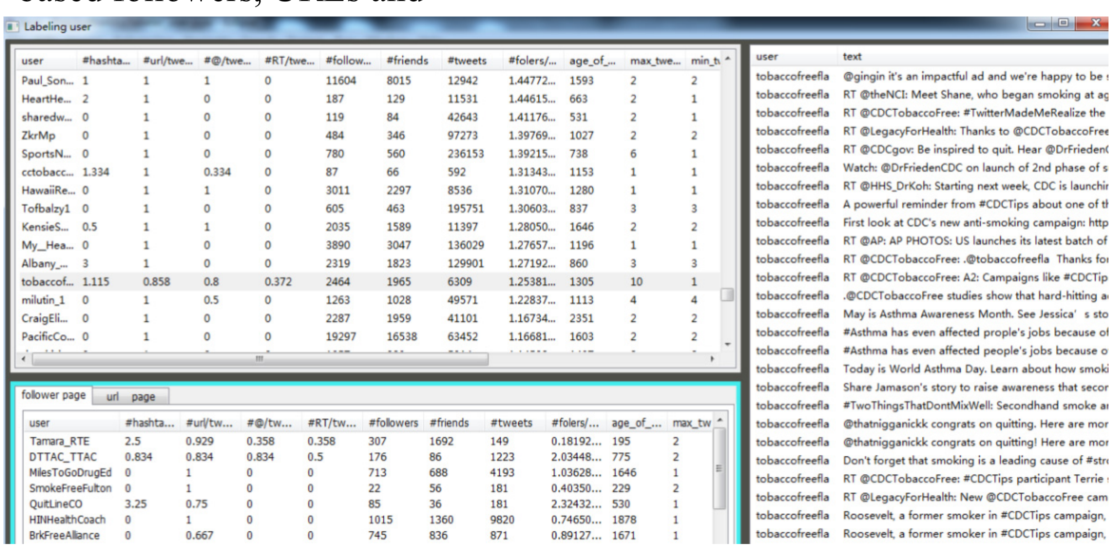
Data Statistics

- Smoking related campaign data from twitter
- Historical tweets obtained from Gnip for user feature completeness
- Center for Disease Control (CDC) launched regulated stop-smoking campaign in US in 2012 and 2013
- E-cig is a commercial campaign and various e-cig brands participated in it

	CDC2012	CDC2013	E-cigarettes
users	3447	7896	3615
tweets	4577	11302	53417
URLs	2262	4481	14730
promoters(labeled)	266	369	612
non-promoters(labeled)	534	431	188

Labeling Camping Promoters

- Labeling decision was made based followers, URLs and



Baselines and Model Variations

- Competitors compared:
  - Local Classifier : Logistic Regression (LR)
  - Iterative Classification Algorithm (ICA)
  - T-MRF (all-nodes, no-priors)
  - T-MRF (user-URL)
  - T-MRF (all-nodes, no-user-user)
  - T-MRF (all)

AUC performance

- Results averaged across 5 disjoint random runs
- T-MRF (all) is consistent in its performance across all thresholds ϵ
- T-MRFs improve over both ICA and Local-LR → Message passing in campaign networks is effective

	CDC2012			**CDC2013**			**E-cigarettes**		
ϵ	0.05	0.10	0.15	0.05	0.10	0.15	0.05	0.10	0.15
Local-LR	0.87	0.87	0.87	0.82	0.82	0.82	0.83	0.83	0.83
ICA	0.88	0.88	0.88	0.86	0.86	0.86	0.84	0.84	0.84
T-MRF(all-nodes,no-priors)	0.83	0.83	0.81	0.73	0.73	0.72	0.68	0.70	0.69
T-MRF(user-url)	0.89	0.89	0.89	0.84	0.85	0.86	0.84	0.84	0.84
T-MRF(all-nodes, no-user-user)	0.88	0.89	0.90	0.88	0.90	0.88	0.86	0.87	0.86
T-MRF(all)	**0.89**	**0.92**	**0.92**	**0.89**	**0.92**	**0.90**	**0.87**	**0.88**	**0.88**

Most tweeted URLS by Promoters and Non-Promoters

- Top/Bottom 10 → Most tweeted URLs by promoters/non-promoters
- For regulated (Govt.) campaigns (e.g., CDC):
-
  - Promoters: news website, government website
  - Non-promoters : links social media including other platforms other than twitter

CDC2012	CDC2013	E-cigarettes
youtube.com	cdc.gov	vaporgod.com
amazon.com	youtube.com	bestcelebrex.blogspot.com
facebook.com	cnn.com	www.shareasale.com
kktv.com	usatoday.com	www.reddit.com
drugstorenews.com	blogs.nytimes.com	www.prweb.com
marketingmagazine.co.uk	medicalnewstoday.com	www.nicotinefreecigarettes.net
adage.com	cbsnews.com	electronicvape.com
cdc.gov	nbcnews.com	youtube.com
howtoquitsmokingfree.com	twitter.com	dfw-ecigs.com
presstitution.com	news.yahoo.com	ecigadvanced.com
youtube.com	twitter.com	purecigs.com
smokefree.gov	cdc.gov	instagram.com
twitlonger.com	youtube.com	houseofelectroniccigarettes.com
cdc.gov	instagram.com	smokelesscigarettesdeals.com
instagram.com	deadspin.com	aan.atrinsic.com
twitpic.com	cnn.com	smokelessdelite.com
tmi.me	soundcloud.com	twitpic.com
facebook.com	usatoday.com	youtube.com
yfrog.com	chacha.com	electroniccigarettesworld.com
chacha.com	huffingtonpost.com	review-electroniccigarette.com

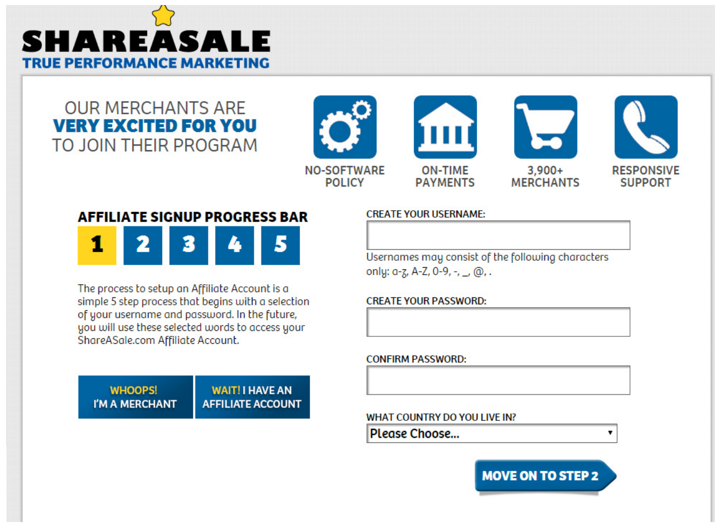
Most tweeted URLS by Promoters and Non-Promoters

- Top/Bottom 10 → Most tweeted URLs by promoters/non-promoters
- For spam/promotion campaigns (e.g., e-cig):
  - Promoters heavily promoted product/e-marketting pages

CDC2012	CDC2013	E-cigarettes
youtube.com	cdc.gov	vaporgod.com
amazon.com	youtube.com	bestcelebrex.blogspot.com
facebook.com	cnn.com	www.shareasale.com
kktv.com	usatoday.com	www.reddit.com
drugstorenews.com	blogs.nytimes.com	www.prweb.com
marketingmagazine.co.uk	medicalnewstoday.com	www.nicotinefreecigarettes.net
adage.com	cbsnews.com	electronicvape.com
cdc.gov	nbcnews.com	youtube.com
howtoquitsmokingfree.com	twitter.com	dfw-ecigs.com
presstitution.com	news.yahoo.com	ecigadvanced.com
youtube.com	twitter.com	purecigs.com
smokefree.gov	cdc.gov	instagram.com
twitlonger.com	youtube.com	houseofelectroniccigarettes.com
cdc.gov	instagram.com	smokelesscigarettesdeals.com
instagram.com	deadspin.com	aan.atrinsic.com
twitpic.com	cnn.com	smokelessdelite.com
tmi.me	soundcloud.com	twitpic.com
facebook.com	usatoday.com	youtube.com
yfrog.com	chacha.com	electroniccigarettesworld.com
chacha.com	huffingtonpost.com	review-electroniccigarette.com

Promoted URL Types

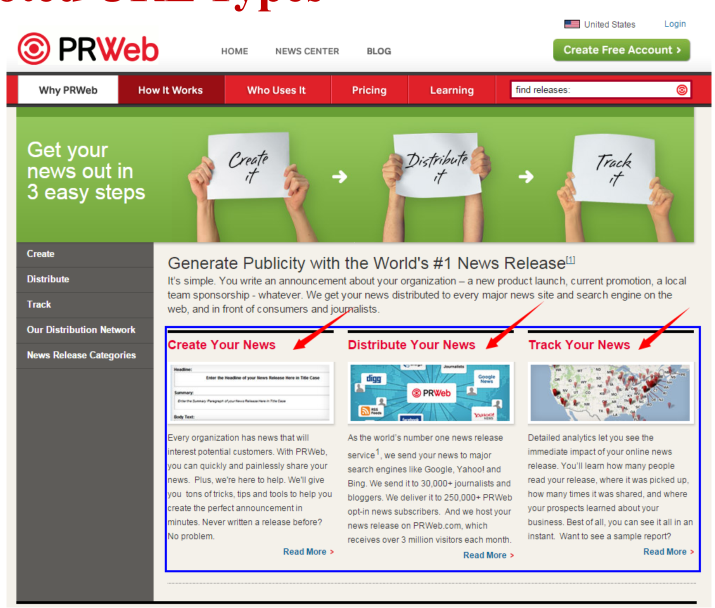
- Affiliate marketing: performance-based marketing in which a business rewards one or more affiliates for each visitor or customer brought by the affiliate's own marketing efforts



Affiliates get paid for the effort to promote merchants

Promoted URL Types

- News/Ad distributors: PRWeb.com

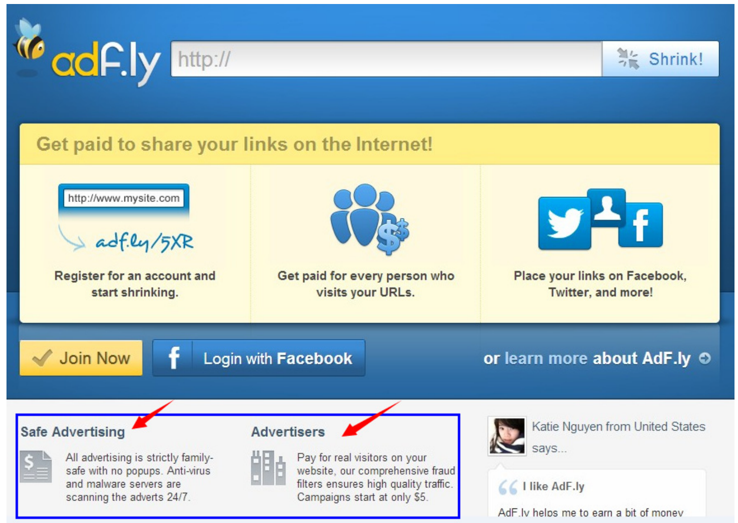


Claim itself as the best e-marketing company that helps distribute news/ad into social media and research engine research

Promoted URL Types

- Paid Link Sharing sites: Ad.fly

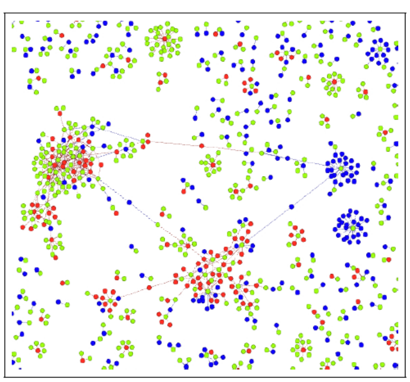
Paid for Mass Self advertising



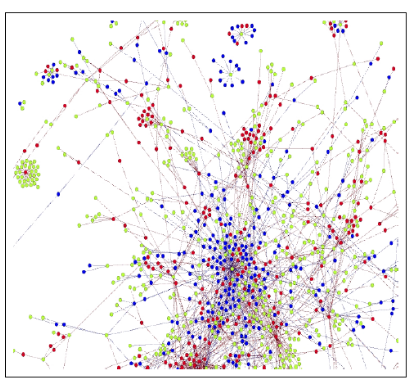
Posterior on Users and URLs

- Portions of network structures for promoters (red), non-promoters (blue) and URLs (green)

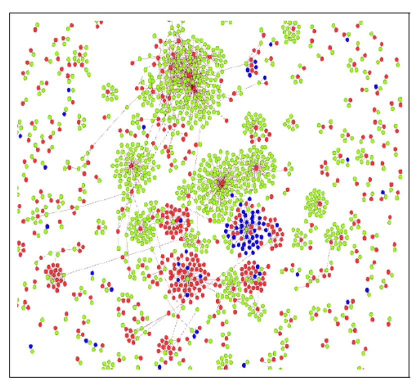
CDC2012



CDC2013



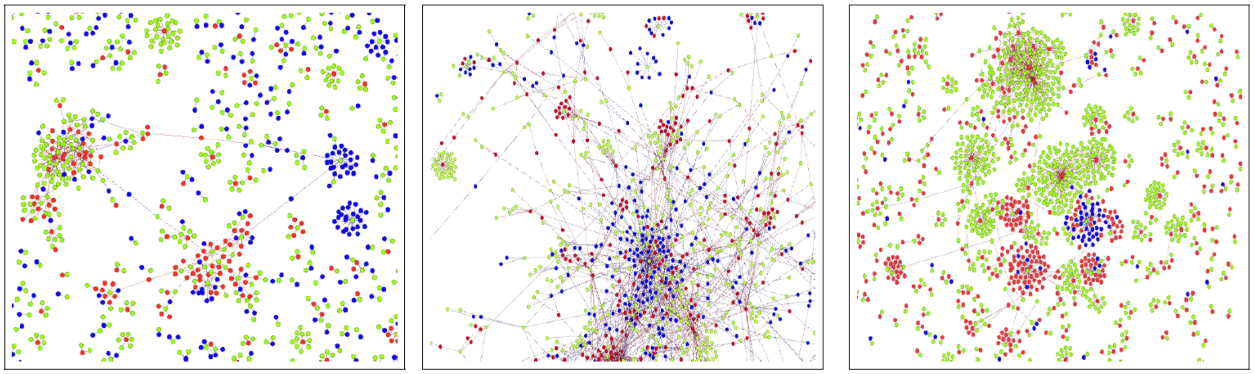
E-cig



Posterior on Users and URLs

- Portions of network structures for promoters (red), non-promoters (blue) and URLs (green)

CDC2012 CDC2013 E-cig

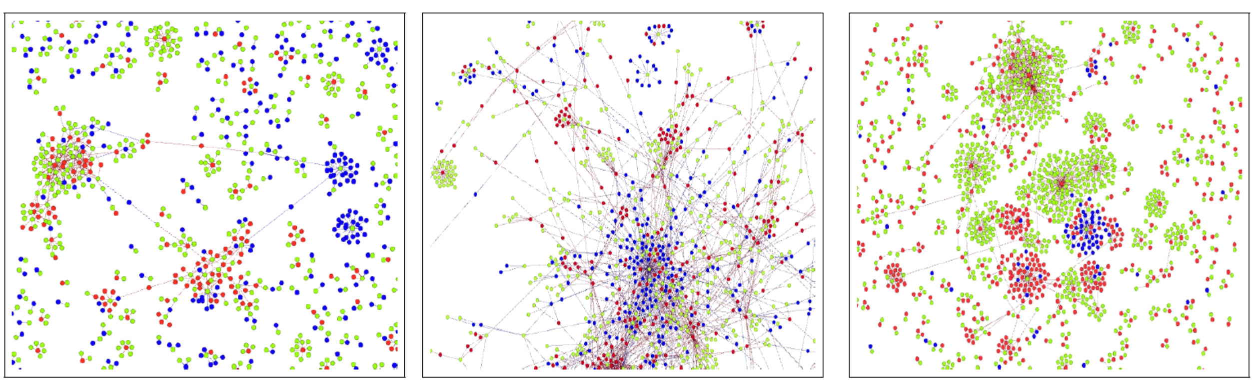


CDC campaigns: non-promoters overlapping with promoters as they show interests in the tweets posted by promoters and proactively join in the discussion

Posterior on Users and URLs

- Portions of network structures for promoters (red), non-promoters (blue) and URLs (green)

CDC2012 CDC2013 E-cig

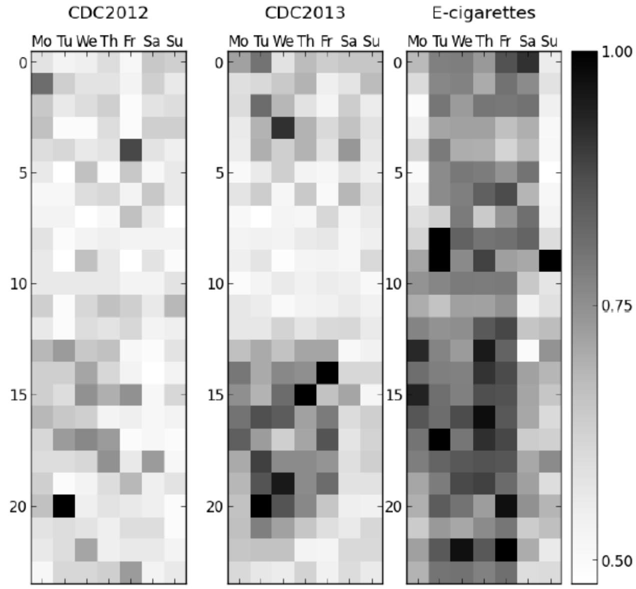


Ecig campaigns: clear separation between promoters and non-promoters as non-promoters tend to be generic users who care less about promoted websites

Posterior on Temporal Activity of Users

- Heat map of posting patterns of promoters on different hours of day and days of week (on GMT Time Zone)

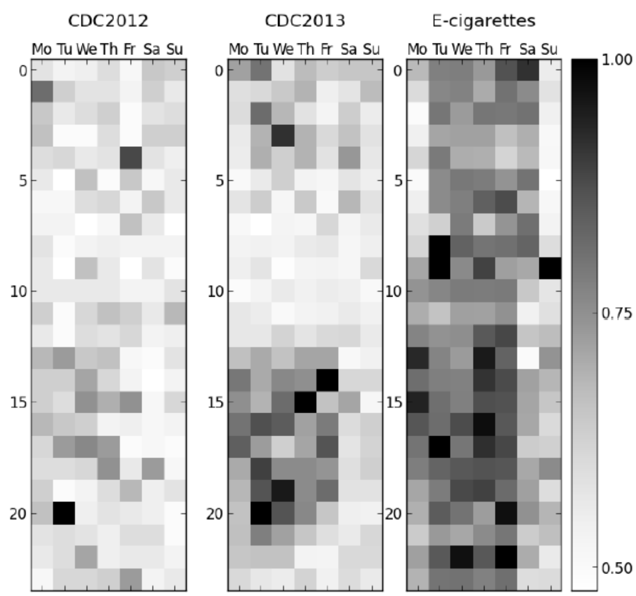
Government campaigns (e.g., CDC) is more regularized and promoters tweet in working hours and weekdays



Posterior on Temporal Activity of Users

- Heat map of posting patterns of promoters on different hours of day and days of week (on GMT Time Zone)

Promoters in commercial campaign restlessly promote their products. Even at night and over weekends, there are still promotional activities which can be attributed to Twitter bots



Variants of LVMs in Deception

- ❑ Tailored topic models for deception [Li et al. ACL 2013]
  - Latent topics can improve opinion spam detection
- ❑ Latent deception prevalence analysis [Ott et al., WWW 2012]
  - "The rate of deception varies according to the costs and benefits of posting fake reviews"
  - Verifiability can lessen deception

Outline

- ❑ Approach#3: Statistical Modeling
- ❑ Approach#4: Authorship/Sockpuppet Detection
  - **P.1: Learning on Similarity Spaces**
  - **P.2: Tri-Training**

Identifying Multi User-Ids

- Use of Multi user-Ids is commonplace in social media
- Key motivations:
  - Use multiple userids to instigate controversy or debates to popularize a topic.
  - Use multiple userids to post fake or deceptive opinions.
- **Q: How to resolve different user-ids (aliases) of the same author?**

Identifying Multi User-Ids

- Multi User-Id identification using linguistic cues [Qian and Liu, EMNLP 2013]
- **Problem Definition:** Given a set of userids $ID = \{id_1, \ldots, id_n\}$ and each id_i has a set of documents Di, we want to identify userids that belong to the same physical author.
- Departure from AA settings : Since some of the userids may belong to the same author, we cannot treat each userid as a class because in that case, we will be classifying based on userids, which won't help us find authors with multiple userids

Identifying Multi User-Ids

- Multi User-Id identification using linguistic cues [Qian and Liu, EMNLP 2013]

- **Problem Definition:** Given a set of userids $ID = \{id_1, \ldots, id_n\}$ and each id_i has a set of documents Di, we want to identify userids that belong to the same physical author.

- Departure from AA settings : Since some of the userids may belong to the same author, we cannot treat each userid as a class because in that case, we will be classifying based on userids, which won't help us find authors with multiple userids

Supervised AA formulation falls short as the goal is to classify actual authors instead of userids

Learning in Similarity Space

- Learn feature associations in the (transformed) similarity space instead of original document space (as in AA) [Qian and Liu, EMNLP 2013]

- Each document d is still represented as a feature vector, but the vector no longer represents the document d itself. Instead, it represents a set of similarities between the document d and a query (document) q.

q: 1:1 2:1 6:2

d1: 1:2 2:1 3:1 d2: 2:2 3:1 5:2

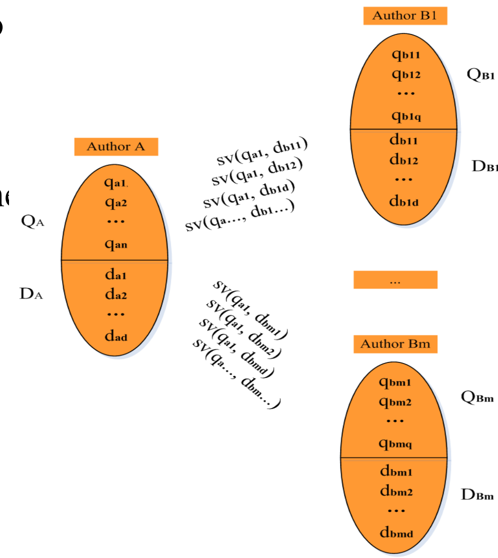
sv (q, d1): +1 1:0.50…

sv (q, d2): -1 1:0.27 …

Similarity can be measured using an s-feature. E.g., cosine: cosine(q, d1) = 0.50 and cosine(q, d2) = 0.27. With more similarity measures more s-features can be produced. The resulting two s-vectors for d1 and d2 with their class labels, 1 (written by author of query q) and -1 (otherwise)

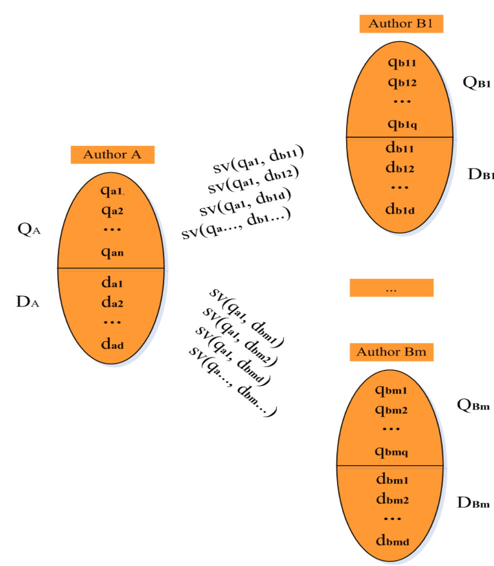
Learning in Similarity Space

- Each author's documents are partitioned into two sets: a query set Q and a sample set D.
- Each d ∈ D and q ∈ Q is represented with a document space vector (d-vector) based on the document itself.
- A similarity vector sv (s-vector) is produced for d. sv consists of a set of similarity values between document d and query q (in a d-vector).



Space Transforming Features

- D-Features: Each feature in the d-vector is a d-feature: Length, Frequency based, TF-IDF based, Richness
- S-Features --- Each feature in the s-vector is a s-feature: Sim-Length, Sim-Retrieval, Sim-Content tf-idf, etc.



Learning Paradigm

- **Candidate identification:** For each userid id_i, we first find the most likely userid id_j ($i \neq j$) that may have the same author as id_i. We call id_j the candidate of id_i. We also call this function candid-iden, i.e., $id_j = candid - iden(id_i)$.

- **Candidate confirmation:** In the reverse order, we apply the function candid-iden on id_j, which produces id_k, i.e., $id_k = candid - iden(id_j)$.

- **Decision making:** If $k = i$, → id_i and id_j are from the same author. Otherwise, id_i and id_j are not from the same author.

Learning Paradigm

- **Candidate identification:** For each userid id_i, we first find the most likely userid id_j ($i \neq j$) that may have the same author as id_i. We call id_j the candidate of id_i. We also call this function candid-iden, i.e., $id_j = candid - iden(id_i)$.

- **Candidate confirmation:** In the reverse order, we apply the function candid-iden on id_j, which produces id_k, i.e., $id_k = candid - iden(id_j)$.

- **Decision making:** If $k = i$, → id_i and id_j are from the same author. Otherwise, id_i and id_j are not from the same author.

worst case time complexity is $O(m^2)$, for a total of m training documents.

In practice, lesser as not all pairwise comparisons are needed. Only a small subset is sufficient (using candidate-iden)

Performance of LSS

- Dataset: 831 reviewers from Amazon.com, 731 for training and 100 for testing; the numbers of reviews in the training and test: 59256 and 14308.
- Baselines: (1) TSL: based on the traditional supervised learning, (2) SimUG/SimAD: uses the word unigrams/all d-features to compare the cosine similarity of queries and samples.

Total # user-ids		10	30	50	80	100
LSS	**Pre**	**100.00**	**100.00**	**100.00**	**100.00**	**98.68**
	Rec	**100.00**	**83.33**	**82.00**	**80.00**	**75.76**
	F1	**100.00**	**90.91**	**90.11**	**88.89**	**85.71**
TSL	**Pre**	**50.00**	**50.00**	**33.33**	**0.00**	**0.00**
	Rec	**11.11**	**3.45**	**2.08**	**0.00**	**0.00**
	F1	**18.18**	**6.45**	**3.92**	**0.00**	**0.00**
SimUG	**Pre**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	Rec	**70.00**	**46.67**	**48.00**	**48.75**	**43.00**
	F1	**82.35**	**63.64**	**64.86**	**65.55**	**60.14**
SimAD	**Pre**	**100.00**	**75.00**	**100.00**	**33.33**	**0.00**
	Rec	**20.00**	**10.35**	**2.00**	**1.28**	**0.00**
	F1	**33.33**	**18.18**	**3.92**	**2.47**	**0.00**

Outline

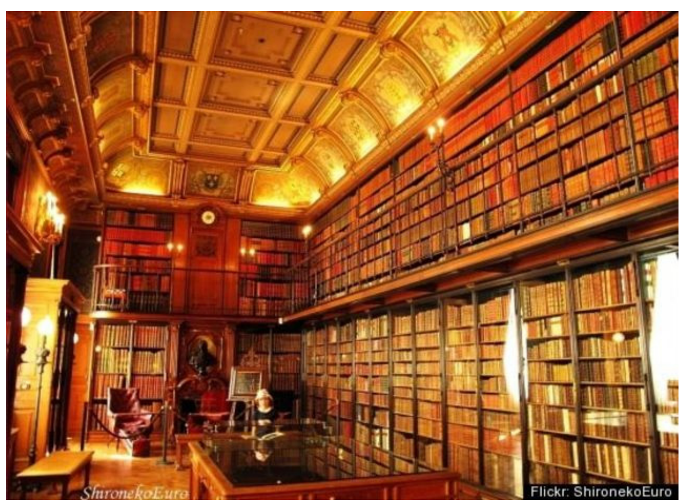
- Approach#3: Statistical Modeling
- Approach#4: Authorship/Sockpuppet Detection
  - **P.1: Learning on Similarity Spaces**
  - **P.2: Tri-Training**

AA with Limited Training Data

- ❑ Authorship Attribution (AA) typically assumes several example documents per author
- ❑ Also traditional AA methods are mostly based on supervised learning.
- ❑ Requirements: for each author, a large number of his/her articles are needed as the training data

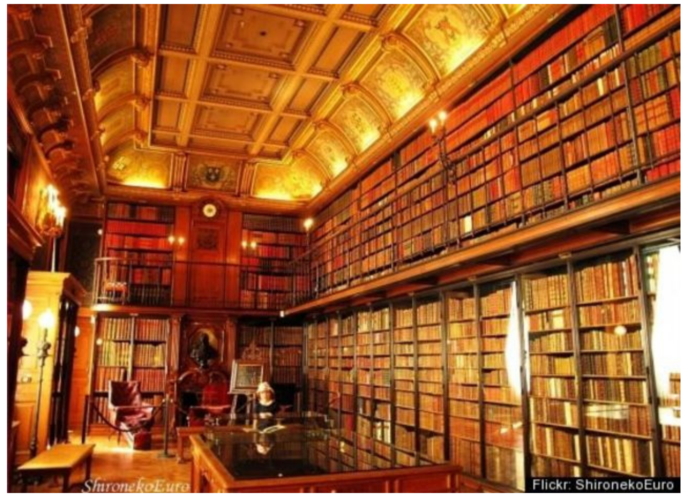
AA with Limited Training Data

How to build reliable AA models with very few labeled examples per author? (e.g., Consumer reviews - a spammer wrote only 3 reviews using an id)



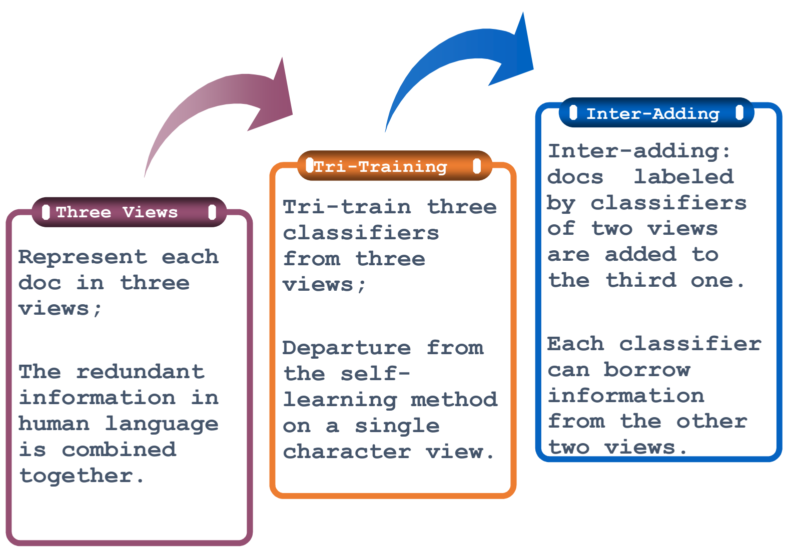
How much data is sufficient?

- 10,000 words per author is regarded as a reasonable training set size [Argamon et al., 2007].
- Dealing with limited data:
  - [Kourtis and Stamatatos, 2011] introduced a variant of the self-training.



Tri-Training

- Tri-Training fro AA using limited training data [Qian et al., ACL 2014].
- Extending classical co-training [Blum and Mitchell, 1998] leveraging sufficient and redundant views on the data



Tri-Training

❑Input:

- A small set of labeled documents $L = \{l_1, \ldots, l_r\}$,
- A large set of unlabeled documents $U = \{u_1, \ldots, u_s\}$,
- A set of test documents $T = \{t_1, \ldots, t_t\}$.

❑Parameters:

- The number of iterations k,
- The size of selected unlabeled documents u,

❑Output:

- t_k's class assignment.

Tri-Training

Add confident examples whenever labels are matched by orthogonal classifiers (i.e., classifiers in two different views)

Extract views L_c, L_l, L_s, U_c, U_l, U_s, T_c, T_l, T_s from L, U, T
Loop for k iterations:
    Randomly select u unlabeled documents U' from U;
    Learn the first view classifier C_1 from L_1 (L_1=L_c, L_l, or L_s);
    Use C_1 to label docs in U' based on U_1(U_1=U_c, U_l, or U_s)
    Learn the second view classifier C_2 from L_2 ($L_2 \neq L_1$)
    Use C_2 to label documents in U' based on U_2 ($U_2 \neq U_1$);
    Learn the third view classifier C_3 from L_3 ($L_2 \neq L_1$, L_2)
    Use C_3 to label documents in U' based on U_3 ($U_2 \neq U_1$, U_2);
    $U_{p1} = \{u \mid u \in U', u.label \text{ by } C_2 = u.label \text{ by } C_3\}$;
    $U_{p2} = \{u \mid u \in U', u.label \text{ by } C_1 = u.label \text{ by } C_3\}$;
    $U_{p3} = \{u \mid u \in U', u.label \text{ by } C_1 = u.label \text{ by } C_2\}$;
    $U = U - U'$, $L_i = L_i \cup U_{pi}$ (i=1..3);
Learn three classifiers C_1, C_2, C_3 from L_1, L_2, L_3;
Use C_i to label t_k in T_i (i=1..3);
Aggregate results from three views

Feature Space

- ❑ Character view → Character n-grams (upto 3-grams)
- ❑ Lexical view → Word n-grams
- ❑ Syntactic view → Content independent structures including n-grams of POS tags (n = 1..3) [Kim et al., SIGIR 2011]

Performance Evaluation of Tri-Training for AA

- ❑ Dataset: 62,000 reviews by 62 users (1,000 reviews per user
- ❑ Data distributions:
  - split each author's documents into three sets:
  - training set: 1%, i.e., 10 docs/author
  - unlabeled set: 79%
  - test set: 20%

Performance Evaluation of Tri-Training for AA

- Baselines

- (1) Self-Training:
  - Using Common N-grams (CNG) + SVM [Kourtis and Stamatatos, 2011] on Char, Lex, Syn views
  - Using LR+SVM on Char, Lex, Syn views

- (2) Co-Training:
  - Using LR on
  - Char + Lex,
  - Char + Syn,
  - Lex + Syn

Performance Evaluation of Tri-Training for AA

- Tri-training outperforms all self-training baselines.

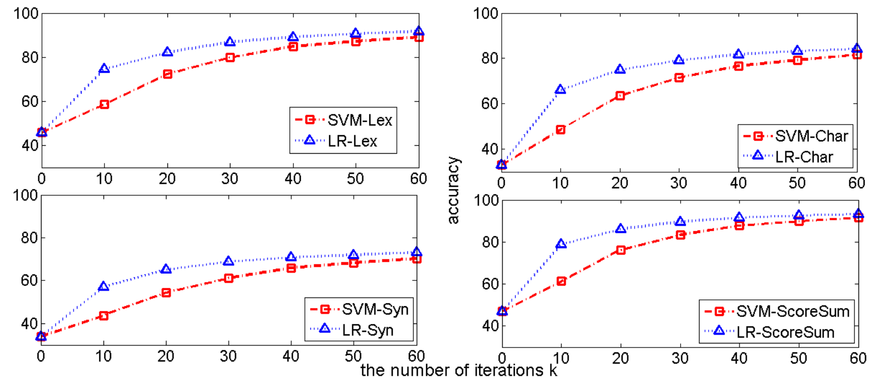
- In tri-training, each individual view may be biased but the views are independent. Then each view is more likely to produce random samples for the other views and thus reduce the bias of each view as the iterations progress

k	Tri Train	SelfTrain:CNG+SVM			SelfTrain:LR+SVM		
		Char	lex	Syn	Char	Lex	Syn
0	46.85	33.22	45.44	34.50	33.22	45.75	34.48
10	78.82	32.47	45.44	34.50	62.56	73.78	51.94
20	86.19	32.47	45.44	34.09	71.21	81.44	59.88
30	89.69	32.47	45.44	34.09	75.21	84.68	63.70
40	91.52	33.69	45.44	34.09	77.46	88.25	65.74
50	92.58	33.69	45.44	34.09	78.64	88.25	67.45
60	93.15	33.69	45.44	34.09	79.54	89.31	68.37

Performance Evaluation of Tri-Training for AA

- Result due to [Qian et al., ACL 2014]
- LR outperforms SVM by a large margin for tri-training when the number of iterations (k) is small.

- One possible reason is that LR is more tolerant to over-fitting caused by the small number of training samples

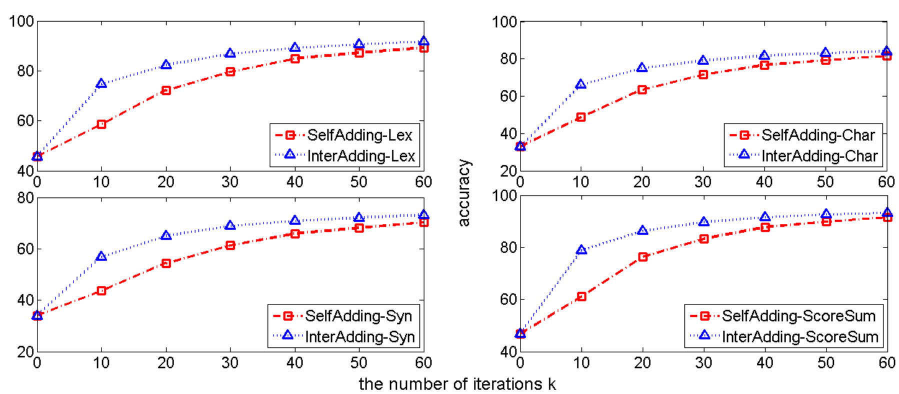


† Contains content originally appearing in [Qian et al., ACL 2014]

Performance Evaluation of Tri-Training for AA

- Result due to [Qian et al., ACL 2014]

- The edge of 3rd view: Adding newly classified samples by two classifiers to the third view improves tri-training.



† Contains content originally appearing in [Qian et al., ACL 2014]

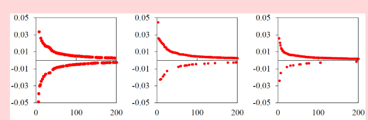
Performance Evaluation of Tri-Training for AA

- ❑ Tri-training outperforms co-training.
- ❑ Consensus predictions by two classifiers are more reliable than those by one classifier.

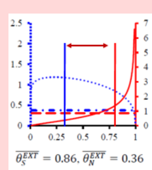
k	Tri Train	Co-Train Char+Lex	Co-Train Char+Syn	Co-Train Lex+Syn
0	46.85	45.75	42.02	45.75
10	78.82	78.84	75.89	78.85
20	86.19	86.02	82.59	85.63
30	89.69	89.32	85.77	88.98
40	91.52	91.14	87.52	91.16
50	92.58	92.19	88.46	92.02
60	93.15	92.81	89.21	92.50

Recap

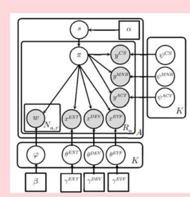
- ❑ Approach#1: Leveraging Linguistic Signals



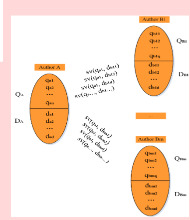
- ❑ Approach#2:Behavioral Modeling



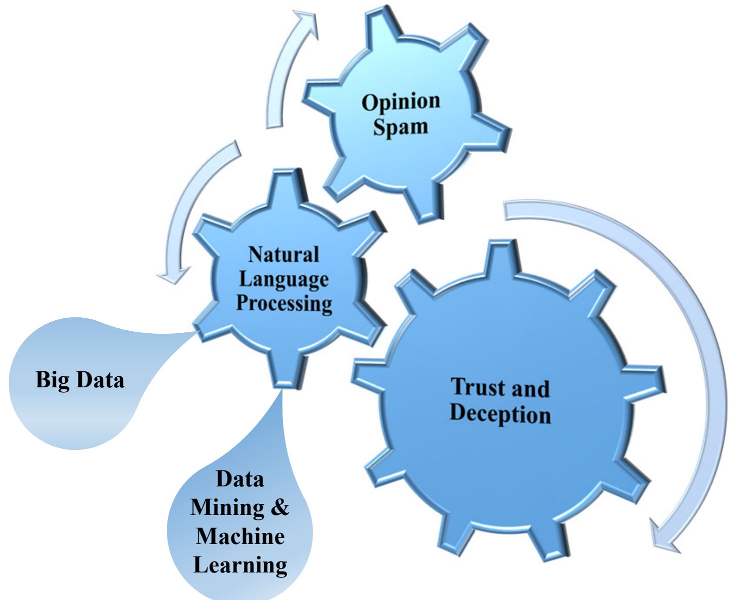
- ❑ Approach#3: Statistical Modeling



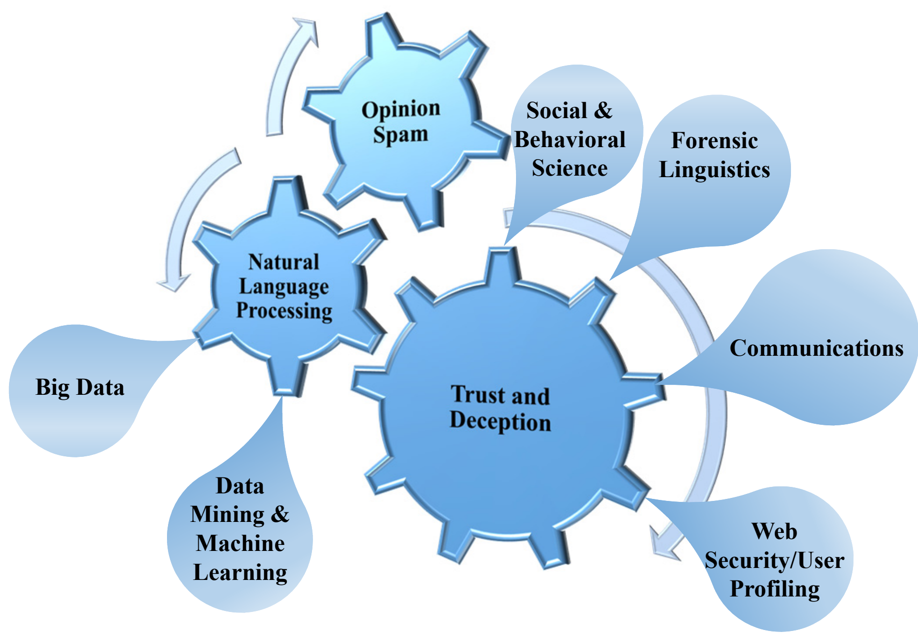
- ❑ Approach#4: Authorship/Sockpuppet Detection



Opinion Spam & Science



Opinion Spam & Science



References

Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. In *Proceedings of the AAAI International Conference on Web and Social Media*.

Shlomo Argamon, Casey Whitelaw, Paul Chase, Sobhan Raj Hota, Navendu Garg, and Shlomo Levitan. 2007. Stylistic text classification using functional lexical features. *Journal of the American Society for Information Science and Technology*, 58(6):802–822.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

Jacob Eisenstein, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. *Empirical Methods in Natural Language Processing*:1277–1287, October.

G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh. 2013. Exploiting Burstiness in Reviews for Review Spammer Detection. *AAAI International Conference on Weblogs and Social Media*.

Song Feng, Longfei Xing, Anupam Gogar, and Yejin Choi. 2012. Distributional Footprints of Deceptive Product Reviews. In *The International AAAI Conference on Weblogs and Social Media*.

References

Y. Feng, S., Banerjee R., Choi. 2012. Syntactic Stylometry for Deception Detection. In Association for Computational Linguistics.

Donato Hernandez Fusilier, Rafael Guzman Cabrera, Manuel Montes-y-Gomez, and Paolo Rosso. 2013. Using PU-Learning to Detect Deceptive Opinion Spam. ACL Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis.

Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity Search in High Dimensions via Hashing. Proceedings of the 25th International Conference on Very Large Data Bases:518–529, September.

Nitin Jindal and Bing Liu. 2007. Review spam detection. In Proceedings of the 16th international conference on World Wide Web - WWW '07, page 1189, New York, New York, USA, May. ACM Press.

Nitin Jindal and Bing Liu. 2008. Opinion Spam and Analysis. In ACM International Conference on Web Search and Data Mining.

Sangkyum Kim, Hyungsul Kim, Tim Weninger, Jiawei Han, and Hyun Duk Kim. 2011. Authorship classification: a discriminative syntactic tree mining approach. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pages 455–464.

References

Mark L Knapp, Roderick P Hart, and Harry S Dennis. 1974. An exploration of deception as a communication construct. Human communication research, 1(1):15–29.

Ioannis Kourtis and Efstathios Stamatatos. 2011. Author identification using semi-supervised learning. In CLEF 2011: Proceedings of the 2011 Conference on Multilingual and Multimodal Information Access Evaluation (Lab and Workshop Notebook Papers), Amsterdam, The Netherlands.

Huayi Li, Zhiyuan Chen, Bing Liu, Xiaokai Wei, and Jidong Shao. 2014a. Spotting Fake Reviews via Collective Positive-Unlabeled Learning. IEEE International Conference on Data Mining.

Huayi Li, Zhiyuan Chen, Arjun Mukherjee, Bing Liu, and Jidong Shao. 2015. Analyzing and Detecting Opinion Spam on a Large-scale Dataset via Temporal and Spatial Patterns. Proceedings of Ninth International AAAI Conference on Web and Social Media.

Huayi Li, Bing Liu, Arjun Mukherjee, and Jidong Shao. 2014b. Spotting Fake Reviews using Positive-Unlabeled Learning. Computaci{ó}n y Sistemas, 18(3).

Huayi Li, Arjun Mukherjee, Bing Liu, Rachel Kornfield, and Sherry Emery. 2014c. Detecting Campaign Promoters on Twitter using Markov Random Field. Proceedings of IEEE International Conference on Data Mining (ICDM 2014).

References

Jiwei Li, Claire Cardie, and Sujian Li. 2013a. TopicSpam: a Topic-Model-Based Approach for Spam Detection. *Annual Meeting of the Association for Computational Linguistics*.

Jiwei Li, Myle Ott, and Claire Cardie. 2013b. Identifying Manipulated Offerings on Review Portals. *Empirical Methods in Natural Language Processing*.

Jiwei Li, Myle Ott, Claire Cardie, and Eduard Hovy. 2014d. Towards a General Rule for Identifying Deceptive Opinion Spam. *Association for Computational Linguistics*.

Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. 2010. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management - CIKM '10*, page 939, New York, New York, USA. ACM Press.

Rada Mihalcea and Carlo Strapparava. 2009. The lie detector: explorations in the automatic recognition of deceptive language. *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*:309–312, August.

A. Mukherjee, B. Liu, J. Wang, N. Glance, and N. Jindal. 2011. Detecting Group Review Spam. In *WWW*.

References

Arjun Mukherjee, V. Venkataraman, B. Liu, and N. Glance. 2013a. Fake Review Detection: Classification and Analysis of Real and Pseudo Reviews. UIC-CS-2013-03.

Arjun Mukherjee, V. Venkataraman, B. Liu, and N. Glance. 2013b. What Yelp Fake Review Filter might be Doing? AAAI International Conference on Weblogs and Social Media.

Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013c. Spotting Opinion Spammers using Behavioral Footprints. Proceedings of the 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.

Arjun Mukherjee, Bing Liu, and Natalie Glance. 2012. Spotting Fake Reviewer Groups in Consumer Reviews. In Proceedings of the 21st international conference on World Wide Web - WWW ’12.

Arjun Mukherjee and Vivek Venkataraman. 2014. Opinion Spam Detection: An Unsupervised Approach using Generative Models. Techincal Report.

J.M. Newman, M.L., Pennebaker, J.W., Berry, D.S., Richards. 2003. Lying words: predicting deception from linguistic styles. Personality and Social Psychology Bulletin:665–675.

References

Myle Ott, Claire Cardie, and Jeff Hancock. 2012. Estimating the prevalence of deception in online review communities. In Proceedings of the 21st international conference on World Wide Web - WWW ’12, page 201, New York, New York, USA. ACM Press.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 2011. Finding Deceptive Opinion Spam by Any Stretch of the Imagination. In Association for Computational Linguistics, pages 309–319.

J.W. Pennebaker, C.K. Chung, M. Ireland, A. Gonzales, and R.J. Booth. 2007. The development and psychometric properties of LIWC2007. LIWC.Net.

Tieyun Qian and Bing Liu. 2013. Identifying Multiple Userids of the Same Author. Empirical Methods in Natural Language Processing.

Tieyun Qian, Bing Liu, Li Chen, and Zhiyong Peng. 2014. Tri-Training for Authorship Attribution with Limited Training Data. Association for Computational Linguistics.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-rad. 2008. Collective Classification in Network Data. Techincal Report:1–24. Belief Propagation in context of Collective Classification.

References

Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2011. Review Graph Based Online Store Review Spammer Detection. 2011 IEEE 11th International Conference on Data Mining:1242–1247, December.

Sihong Xie, Guan Wang, Shuyang Lin, and Philip S. Yu. 2012. Review spam detection via temporal pattern discovery. Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’12:823.

Junting Ye and Leman Akoglu. 2015. Discovering Opinion Spammer Groups by Network Footprints. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases.