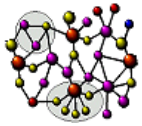


Center for Web Intelligence
School of CTI, DePaul University
Chicago, Illinois, USA

The Impact of Attack Profile Classification on the Robustness of Collaborative Recommendation*

Chad Williams, Runa Bhaumik, Robin Burke, *Bamshad Mobasher*
Center for Web Intelligence
School of Computer Science, Telecommunication, and Information Systems
DePaul University, Chicago, Illinois, USA

WebKDD 2006
Philadelphia, PA

* Supported in part by the NSF Cyber Trust grant IIS-0430303

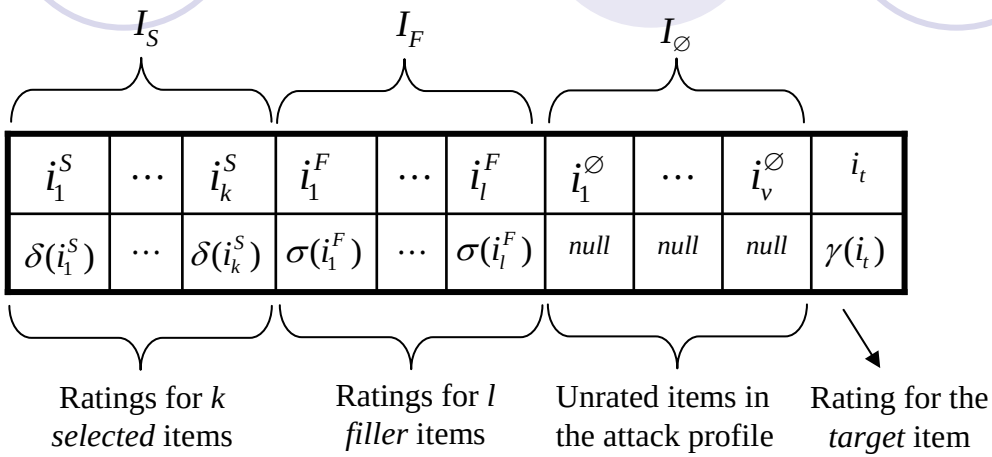
Outline

- Vulnerabilities in collaborative recommendation
  - Profile injection attacks
  - Basic attack models
- Detection and Response
  - A Classification approach to detection
  - Generic and model-specific attributes
- Results
  - Effectiveness of detection
  - Impact of detection on system robustness

Profile Injection Attacks

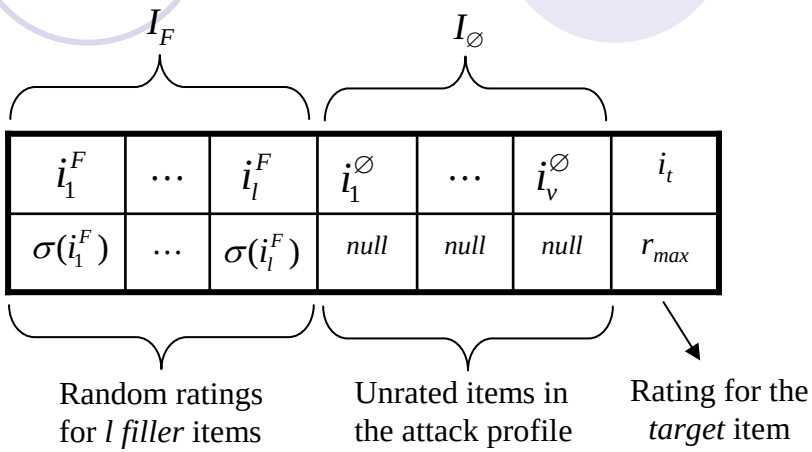
- **Consist of a number of "attack profiles"**
  - added to the system by providing ratings for various items
  - engineered to bias the system's recommendations
  - Two basic types:
    - "Push attack" ("Shilling"): designed to promote an item
    - "Nuke attack": designed to demote a item
  - Prior work has shown that CF recommender systems are highly vulnerable to such attacks
- **Attack Models**
  - strategies for assigning ratings to items based on knowledge of the system, products, or users
  - examples of attack models: "random", "average", "bandwagon", "segment", "love-hate"

A Generic Attack Profile



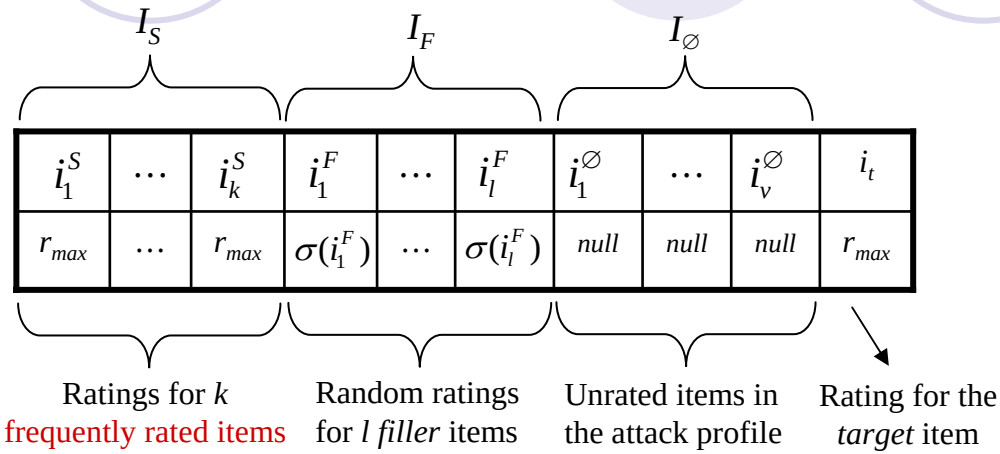
- Previous work considered simple attack profiles:
  - No selected items, i.e., $I_S = \varnothing$
  - No unrated items, i.e., $I_\varnothing = \varnothing$
  - Attack models differ based on ratings assigned to filler and selected items

Average and Random Attack Models



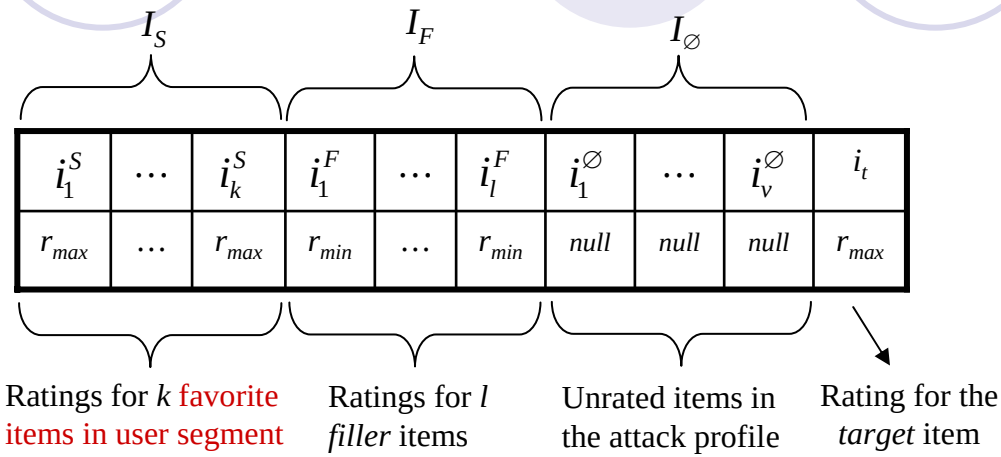
- **Random Attack**: filler items are assigned random ratings drawn from the overall distribution of ratings *on all items* across the whole DB
- **Average Attack**: ratings each filler item drawn from distribution defined by average rating for *that item* in the DB
- The percentage of filler items determines the amount knowledge (and effort) required by the attacker

Bandwagon Attack Model



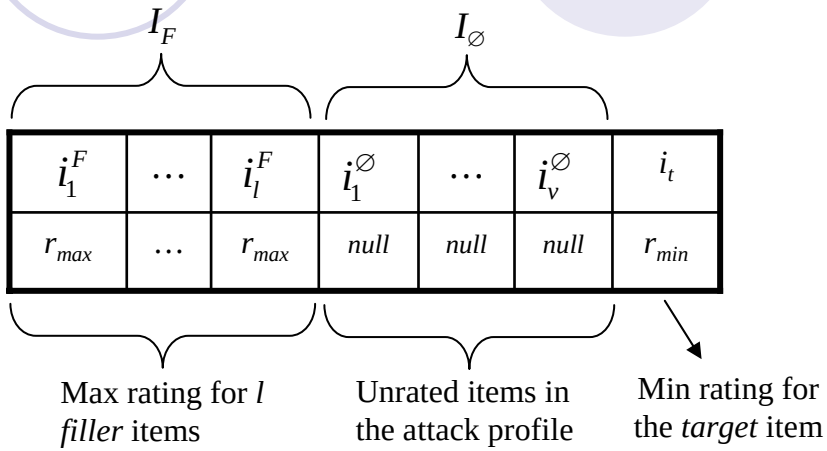
- What if the system's rating distribution is unknown?
  - Identify products that are frequently rated (e.g., “blockbuster” movies)
  - Associate the pushed product with them
  - Ratings for the filler items centered on overall system average rating (Similar to Random attack)
  - frequently rated items can be guessed or obtained externally

Segment Attack Model



- Assume attacker wants to push product to a target segment of users
  - those with preference for similar products
    - fans of Harrison Ford
    - fans of horror movies
  - like bandwagon but for semantically-similar items
  - originally designed for attacking item-based CF algorithms
    - maximize *sim*(target item, segment items)
    - minimize *sim*(target item, non-segment items)

Nuke Attacks: Love/Hate Attack Model



- A limited-knowledge attack in its simplest form
  - Target item given the minimum rating value
  - All other ratings in the filler item set are given the maximum rating value
- Note:
  - Variations of this (an the other models) can also be used as a push or nuke attacks, essentially by switching the roles of r_{min} and r_{max}.

Defense Against Attacks

- **Profile Classification**
  - Automatically identify attack profiles and exclude them from predictions
  - Reverse-engineered profiles likely to be most damaging
  - Increase cost of attacks by detecting most effective attacks
  - Characteristics of known attack models are likely to appear in other effective attacks as well
- **Basic Approach**
  - Create attributes that capture characteristics of suspicious profiles
  - Use attributes to build classification models
  - Apply model to user profiles to identify and discount potential attacks
- **Two Types of Detection Attributes**
  - **Generic** – Focus on overall profile characteristics
  - **Model-specific** – based on characteristics of specific attack models
    - Partition profile to maximize similarity to known models
    - Generate attributes related to partition characteristics

Attributes for Profile Classification

- **Why detection attributes?**
  - Reduce dimensions
  - Generalize profile signatures to make training practical
    - Train for characteristics of an attack,
    - Rather than train for attack on item X

	Item 1	Item 2	Item 3												Item N
Profile 1	4		2						…		3				
Profile 2		5					2		…						4

	Attr 1	Attr 2	Attr 3	…	Attr M
Profile 1	.65	.45	.12	…	.72
Profile 2	.78	.23	.13	…	.98

In our case reducing from 1682 dimensions to 15

- **Two Types of Detection Attributes**
  - **Generic** - focus on overall profile characteristics
  - **Model-specific** – based on characteristics of specific attack models

Examples of Generic Attributes

- Weighted Deviation from Mean Agreement (WDMA)
  - Average difference in profile's rating from mean rating on each item weighted by the item's inverse rating frequency squared
- Weighted Degree of Agreement (WDA)
  - Sum of profile's rating agreement with mean rating on each item weighted by inverse rating frequency
- Average correlation of the profile's *k* nearest neighbors
  - Captures rogue profiles that are part of large attacks with similar characteristics
- Variance in the number of ratings in a profile compared to the average number of ratings per user
  - Few real users rate a large # of items

$$\mathrm{WDMA}_u = \frac{\sum_{i=0}^{n_u} \frac{\left| r_{u,i} - \overline{r_i} \right|}{l_i^2}}{n_u}$$

$$\mathrm{WDA}_u = \sum_{i=0}^{n_u} \frac{\left| r_{u,i} - \overline{r_i} \right|}{l_i}$$

$$\mathrm{DegSim}_j = \frac{\sum_{i=1}^{k} W_{ij}}{k}$$

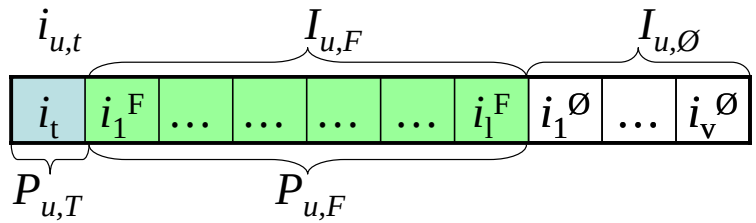
$$\mathrm{LengthVar}_j = \frac{\left| \#ratings_j - \overline{\#ratings} \right|}{\sum_{i=0}^{N} (\#ratings_j - \overline{\#ratings})^2}$$

Model Specific Attributes

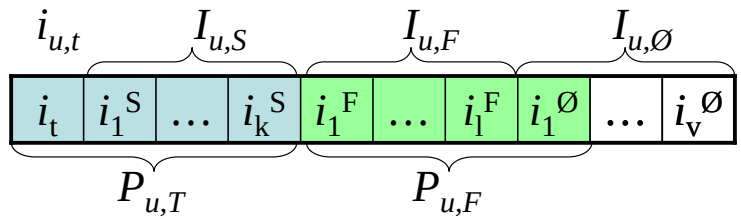
- Partition profile to maximize similarity to known models
- Generate attributes related to partition characteristics that would stand out if the profile was that type of attack

Examples of Model Specific Attributes

- **Average attack detection model**
  - Partition profile to minimize variance in ratings in $P_{u,F}$ from mean rating for each item
  - For average attack, the mean variance of the filler partition is likely less than an authentic user
- **Segment attack detection model**
  - Partition profile into items with high ratings and low ratings
  - For segment attack, the difference between the average rating of these two groups is likely greater than that of an authentic user
- **Target focus detection model (TMF)**
  - Use the identified $P_{u,T}$ partitions to identify concentrations of items under attack across all profiles



$$\text{MeanVar}(r_{target}, j) = \frac{\sum_{i \in (P_j - r_{target})} \left(r_{i,j} - \bar{r}_i\right)^2}{|K|}$$



$$\text{FMTD}_u = \left| \left(\frac{\sum_{i \in P_{u,T}} r_{u,i}}{|P_{u,T}|} \right) - \left(\frac{\sum_{k \in P_{u,F}} r_{u,k}}{|P_{u,F}|} \right) \right|$$

Methodological Note

- Data set
  - Using MovieLens 100K data set
  - Data split 50% training, 50% test
- Profile classifier - Supervised training approach
  - *k*NN classifier, *k*=9
  - Training data
    - Half of actual data labeled as “Authentic”
    - Insert a mix of attack profiles built from several attack models labeled as “Attack”
  - Test data
    - Start with second half of actual data
    - Insert test attack profiles targeting different movies than targeted in training data
- Recommendation Algorithm
  - User based *k*NN, *k* = 20
- Evaluating results
  - 50 different target movies
    - selected randomly but mirroring overall distribution
  - 50 users randomly pre-selected
    - Results were averaged over all runs for each movie-user pair

Evaluation Metrics

Detection attribute value:

- Information Gain – attack profile vs. authentic profile

Classification performance:

True positive = # of attack profiles correctly identified

False positive = # of authentic profiles misclassified as attacks

False negatives = # of attack profiles misclassified as authentic

- *Precision* = true positives / (true pos. + false pos.)

  Percent of profiles identified as attacks that are attacks

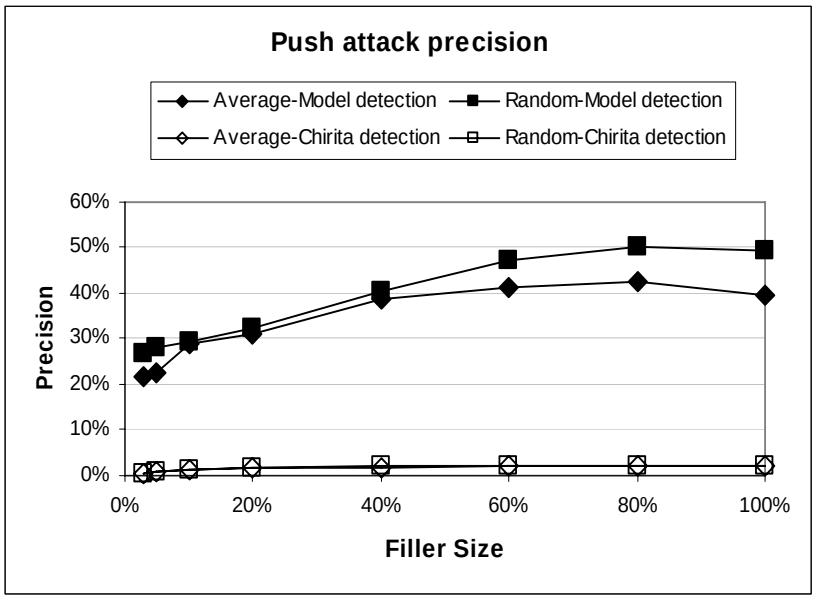
- *Recall* = true positives / (true pos. + false negatives)

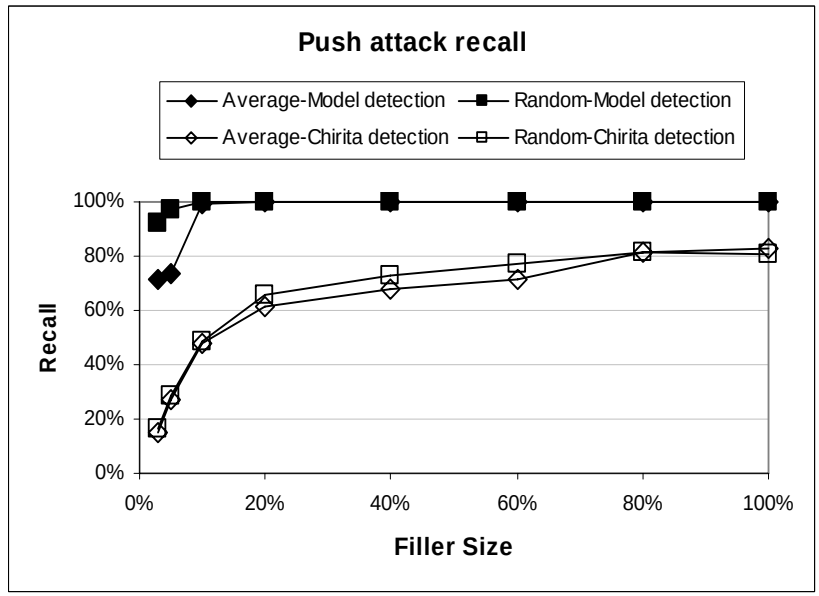
  Percent of attack profiles that were identified correctly

Recommender robustness:

- *Prediction shift* – change in recommender's prediction resulting from the attack

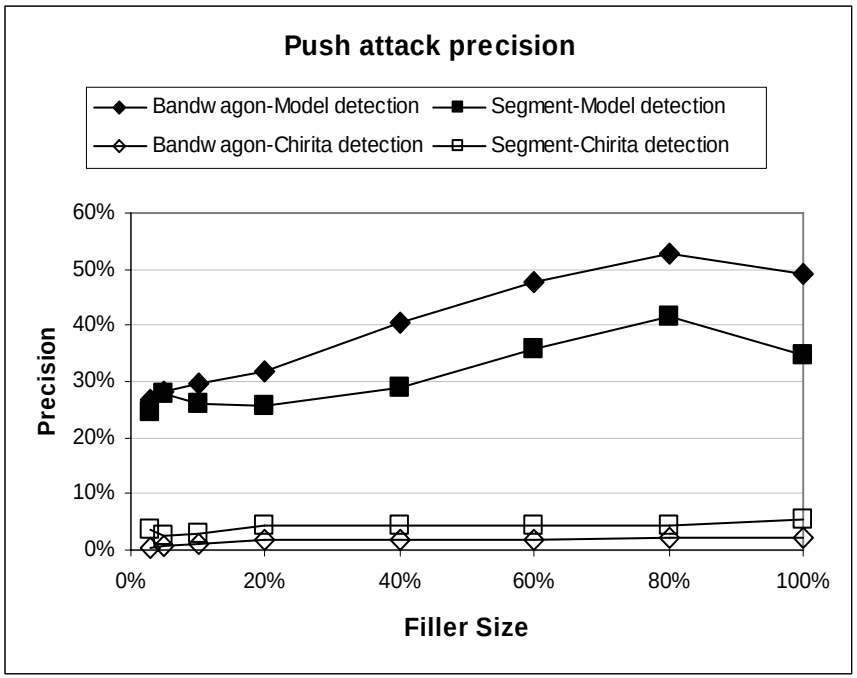
Classification Effectiveness:
Average and Random Push Attacks

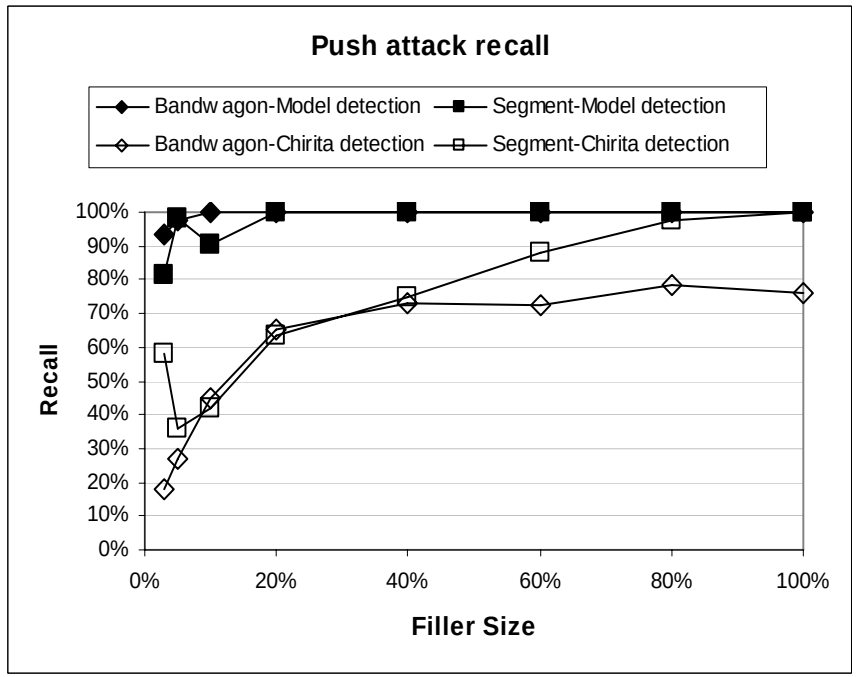




Note: As a baseline we compared our classifier with the ad hoc approach for attack detection by Chirita et al., WIDM 2005, which does not use all of the proposed attributes and does not build a classification model.

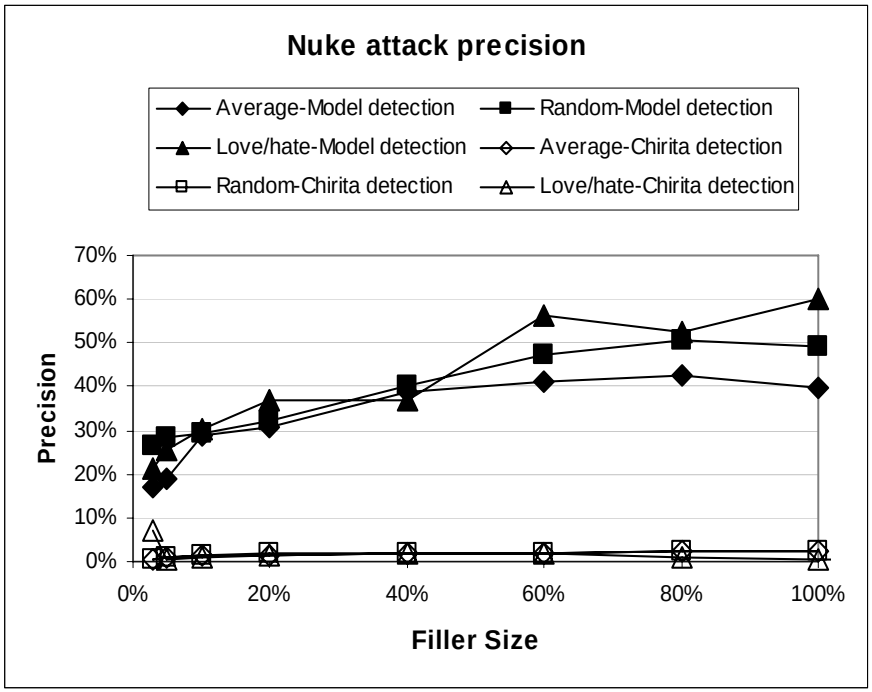
Classification Effectiveness:
Bandwagon and Segment Push Attacks

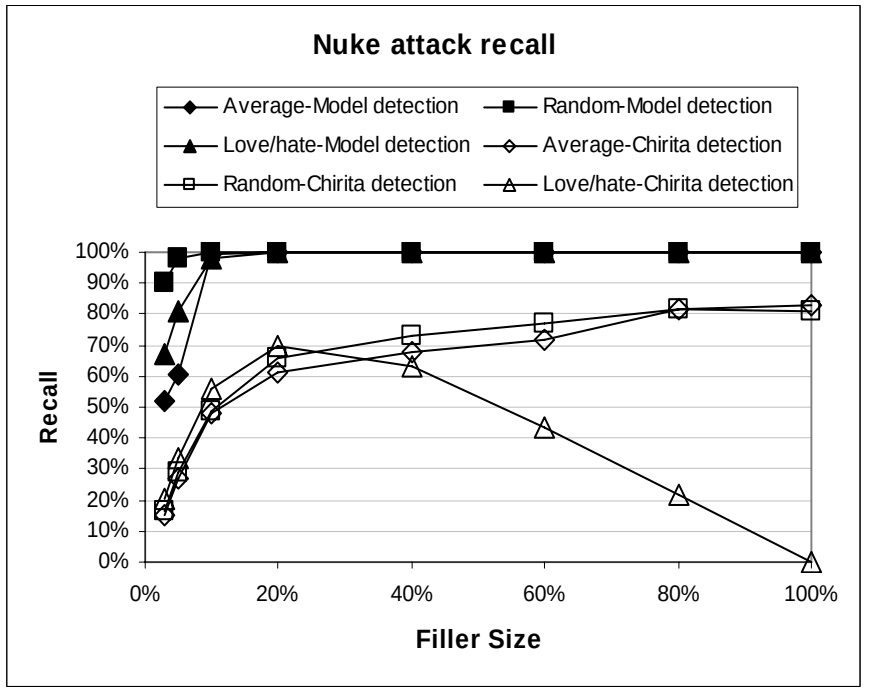




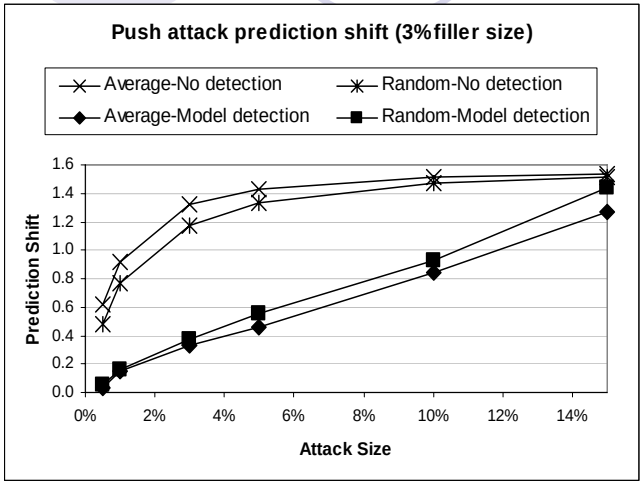
Classification Effectiveness:

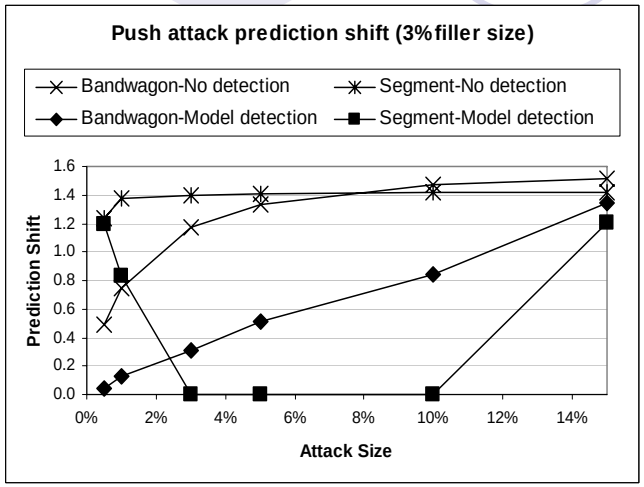
Nuke Attacks: Average, Random, Love/Hate

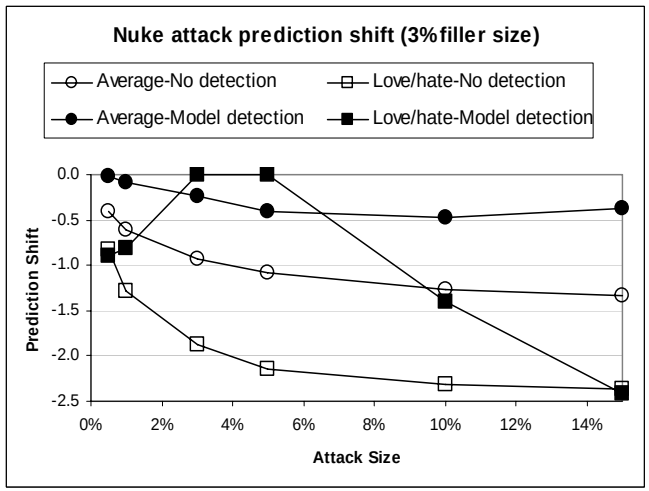




Robustness:
Impact of Detection on Prediction Shift Due to Attacks







Conclusions

- Collaborative spam (clam?)
  - Worse than we thought; common algorithms vulnerable; targeting quite easy to achieve
  - Attacks, if designed correctly, can require very limited system-specific knowledge
  - Need methods to detect and neutralize attacks
- Understanding properties of attack models
  - Can help in designing more robust algorithms
  - Needed to develop effective detection and response algorithms

Future Work

- Detection and Response
  - Develop and a comprehensive D&R framework which combines anomaly detection methods and profile classification approaches
  - Profile classification: explore “obfuscated” attacks
  - More robust hybrid and model-based algorithms
- Other Future Work
  - Explore vulnerabilities of other recommendation algorithms
  - Clickstream data sets: Web usage data
  - Text-based data sets: user opinions, e.g., CNet

Questions

?

Informativeness of Attributes

Table 1: Information gain for the detection attributes against push attacks.

	Attribute	Random		Average		Bandwagon		Segment	
		Info Gain	Rank	Info Gain	Rank	Info Gain	Rank	Info Gain	Rank
Generic	DegSim (k = 450)	0.161	6	0.116	9	0.180	5	0.180	12
	DegSim' (k = 2, d = 963)	0.103	9	0.177	7	0.101	9	0.213	10
	WDA	0.233	4	0.229	3	0.234	4	0.246	5
	LengthVariance	0.267	1	0.267	1	0.267	1	0.269	3
	WDMA	0.248	2	0.238	2	0.248	2	0.229	8
	RDMA	0.240	3	0.229	4	0.240	3	0.239	7
Model-specific	FillerMeanDiff*	0.064	13	0.084	13	0.064	13	0.244	6
	MeanVar*	0.099	10	0.093	12	0.100	10	0.222	9
	ProfileVariance*	0.083	12	0.109	10	0.086	12	0.274	2
	FMTD*	0.130	7	0.189	5	0.131	7	0.276	1
	FMV*	0.094	11	0.126	8	0.095	11	0.263	4
	TMF*	0.194	5	0.185	6	0.174	6	0.176	13

Table 2: Information gain for the detection attributes against nuke attacks.

Attribute	Random		Average		Love/Hate	
	Info Gain	Rank	Info Gain	Rank	Info Gain	Rank
DegSim (k = 450)	0.161	6	0.111	10	0.155	11
DegSim' (k = 2, d = 963)	0.104	10	0.176	5	0.213	9
WDA	0.234	4	0.229	4	0.253	5
LengthVariance	0.267	1	0.267	1	0.267	3
WDMA	0.248	2	0.238	2	0.244	8
RDMA	0.240	3	0.229	3	0.249	7
FillerMeanDiff*	0.084	12	0.094	12	0.249	6
MeanVar*	0.109	9	0.103	11	0.200	10
ProfileVariance*	0.095	11	0.121	8	0.095	12
FMTD*	0.138	8	0.154	7	0.276	1
FMV*	0.077	13	0.069	13	0.276	1
TMF*	0.190	5	0.162	6	0.267	4

A Successful Push Attack

	Item1	Item 2	Item 3	Item 4	Item 5	Item 6	Correlation with Alice
Alice	5	2	3	3		?	
User 1	2		4		4	1	-1.00
User 2	2	1	3		1	2	0.33
User 3	4	2	3	2		1	.90
User 4	3	3	2		3	1	0.19
User 5		3		2	2	2	-1.00
User 6	5	3		1	3	2	0.65
User 7		5		1	5	1	
Attack 1	**2**		**3**		**2**	**3**	**-1.00**
Attack 2	**3**	**2**	**3**		**2**	**5**	**0.76**
Attack 3	**3**	**2**	**2**	**2**		**5**	**0.93**

Prediction

Best Match

"user-based" algorithm using k-nearest neighbor with k = 1