Learning-based Approaches for Link Discovery Given Unlabeled Data (KDD 2013)

Shou-De Lin
PI in Machine Discovery and Social Network Mining Lab
National Taiwan University
sdlin@csie.ntu.edu.tw

Link Discovery On Networks

- Goal: predicting the existence and type of links between two entities

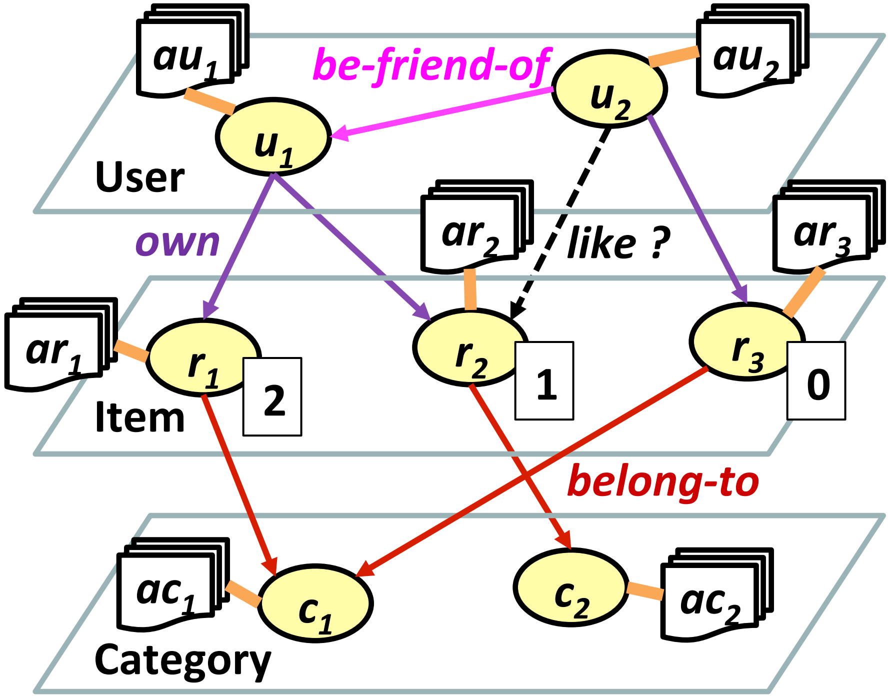
- Supervised systems can be built with labeled data

- Sometimes, links to be discovered are unlabeled in training
  - **Eg**. predict whether a user will "*like*" a post in Foursquare
    - The "*like*" relationship has not been labeled due to privacy concern

- Most literatures do not handle such problem

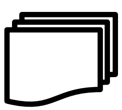
Problem and Motivation

- Individual opinion (ex. customer's preference) is valuable
  - but sometimes concealed due to privacy (ex. Foursquare "*like*")
  - Fortunately, aggregative statistics (total count) is usually available
- Goal: Predict unlabeled relationship (or unseen link) using
  - Heterogeneous social network info
  - Attributes of nodes
  - Aggregative statistics

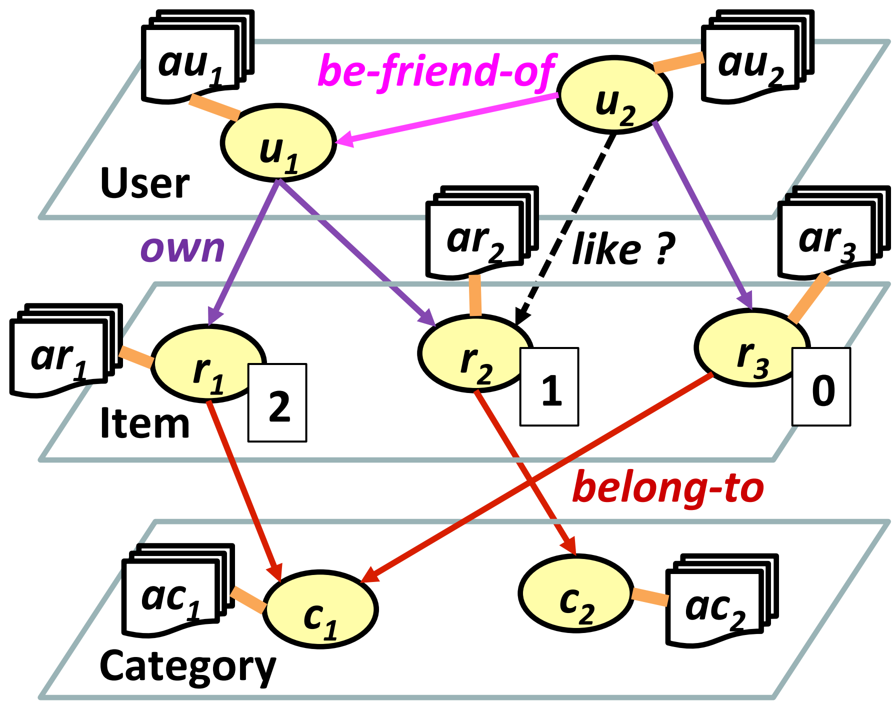


Challenges

- Diverse information exists
- Lack of labeled data
  - With labeled data we can directly perform supervised learning (ex. predicting "*own*"), but without?

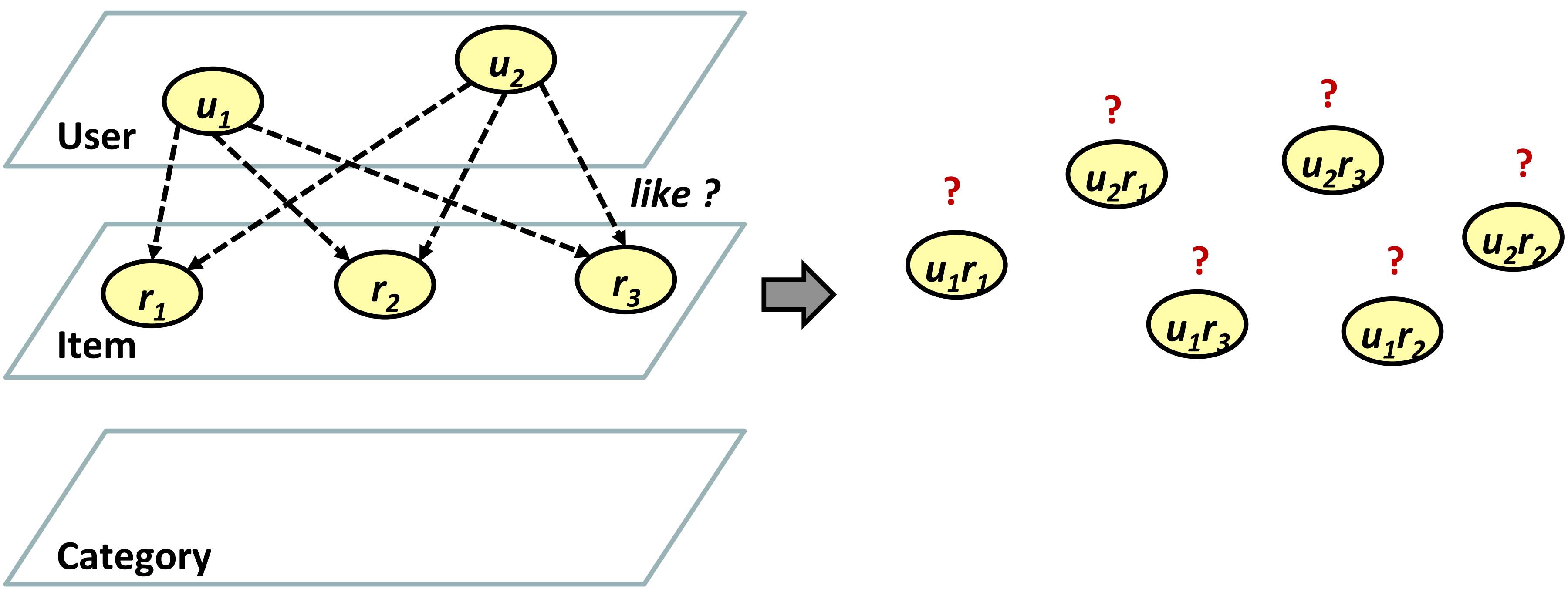


We omit attributes of nodes (ex. number of friends of u_1) for brevity



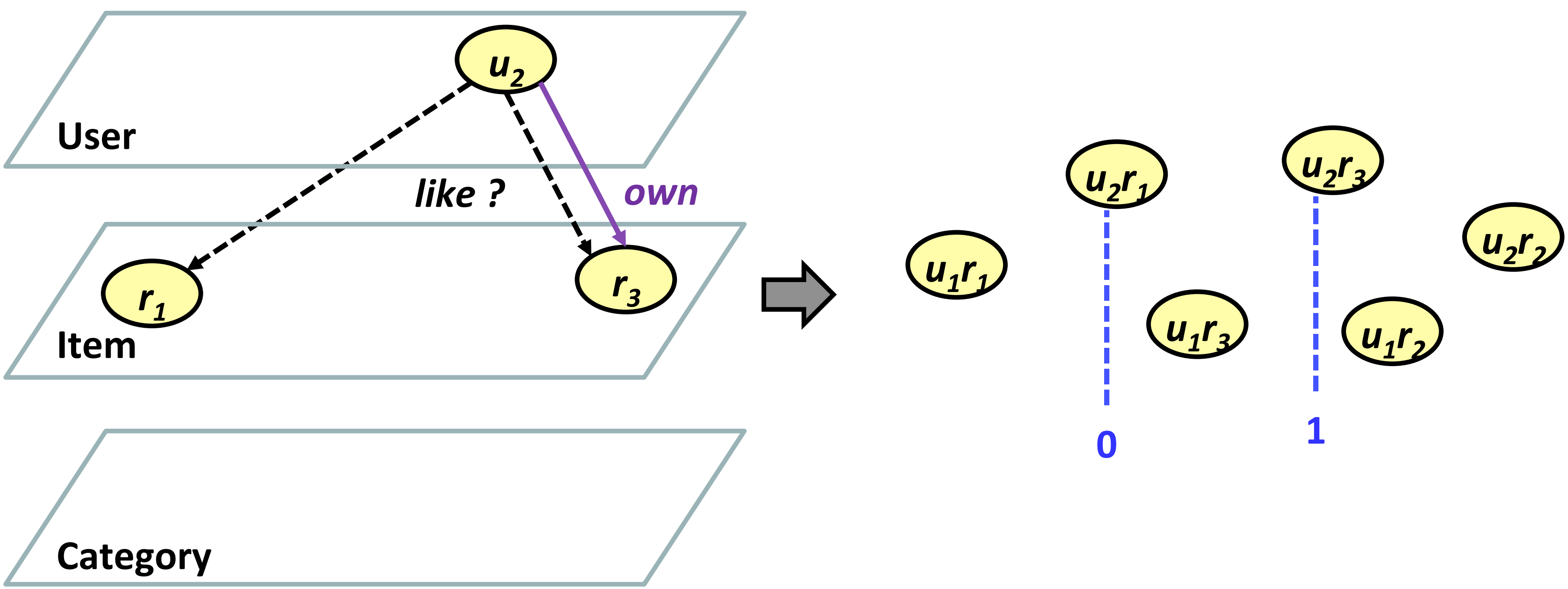
Search Space

- Intuitively, we can enumerate all possible candidate pairs
  - E.g. Assume 2 users, 3 items, then there are totally 2 * 3 = 6 possible links (user-item pairs)
  - The size of search space is $2^6 = 64$ combinations
  - Our goal is to estimate probabilities of these 6 links



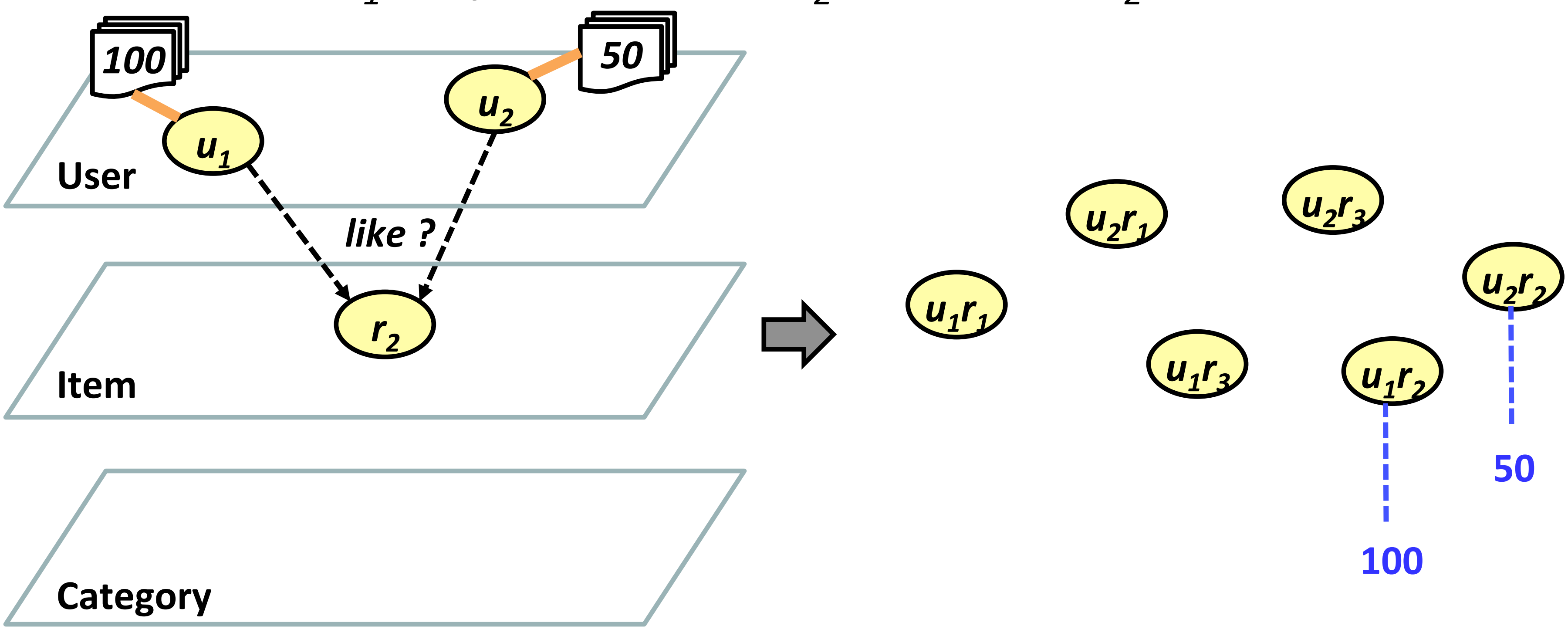
Intuition 1: Simple Heuristics

- There are some knowledge about the 'link' relationship we can exploit
- Model the characteristics of the candidate pairs
  - Ex. **S1**: people tend to like their own items, or vice versa
  - That is, u_2 tends to like r_3 more than like r_1



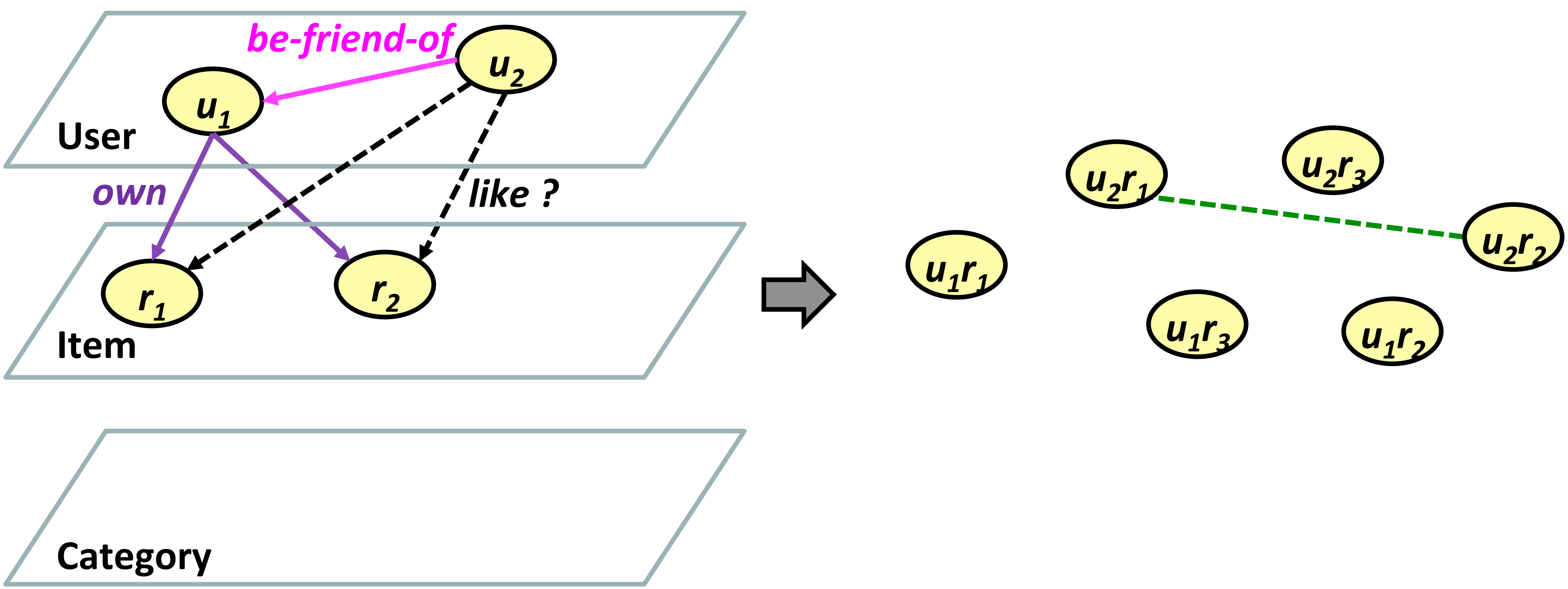
Intuition 2: Simple Heuristics (cont.)

- Other simple heuristics may be applied
  - Ex. **S2**: people with more friends have higher tendency to like items
  - Suppose u_1 has 100 friends, and u_2 has 50
  - That is, u_1 may tends to like r_2 more than u_2



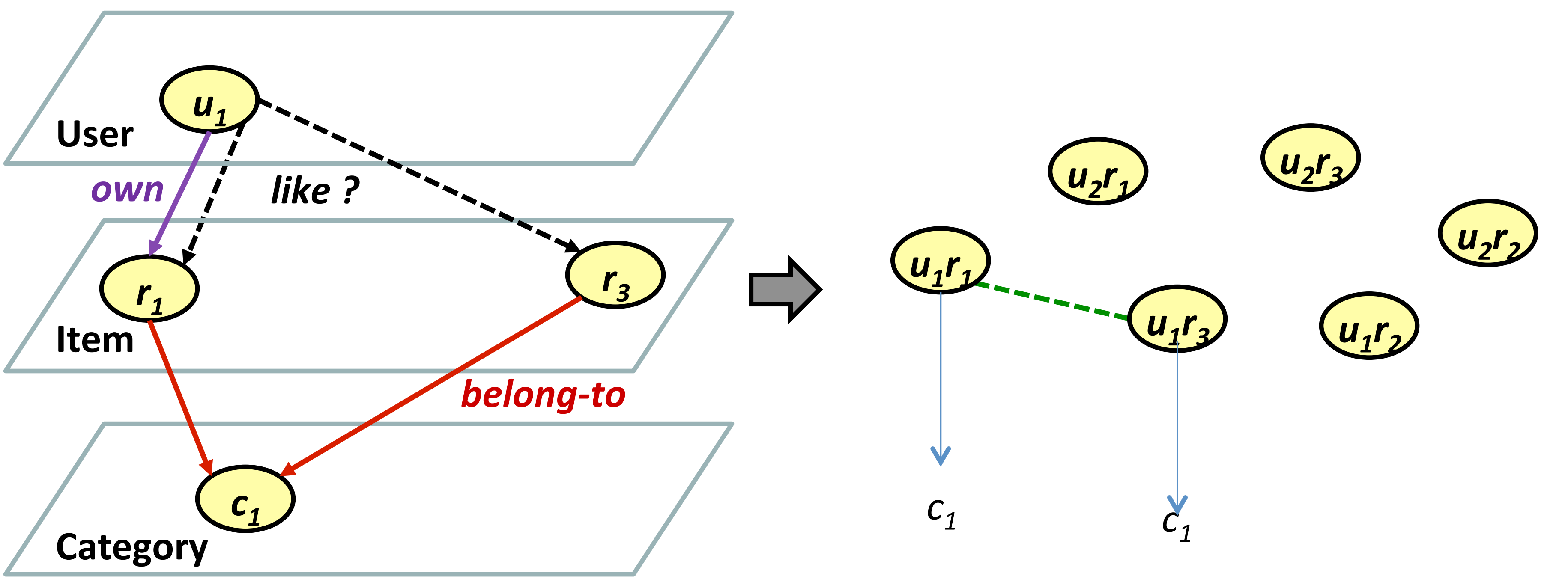
Intuition 3: Complex Heuristics

- Model the relations of the candidate pairs
  - Ex. **C1**: people tend to like social neighbors' items in similar extend
  - That is, if u_2 like friend's item r_1, he/she may also like r_2



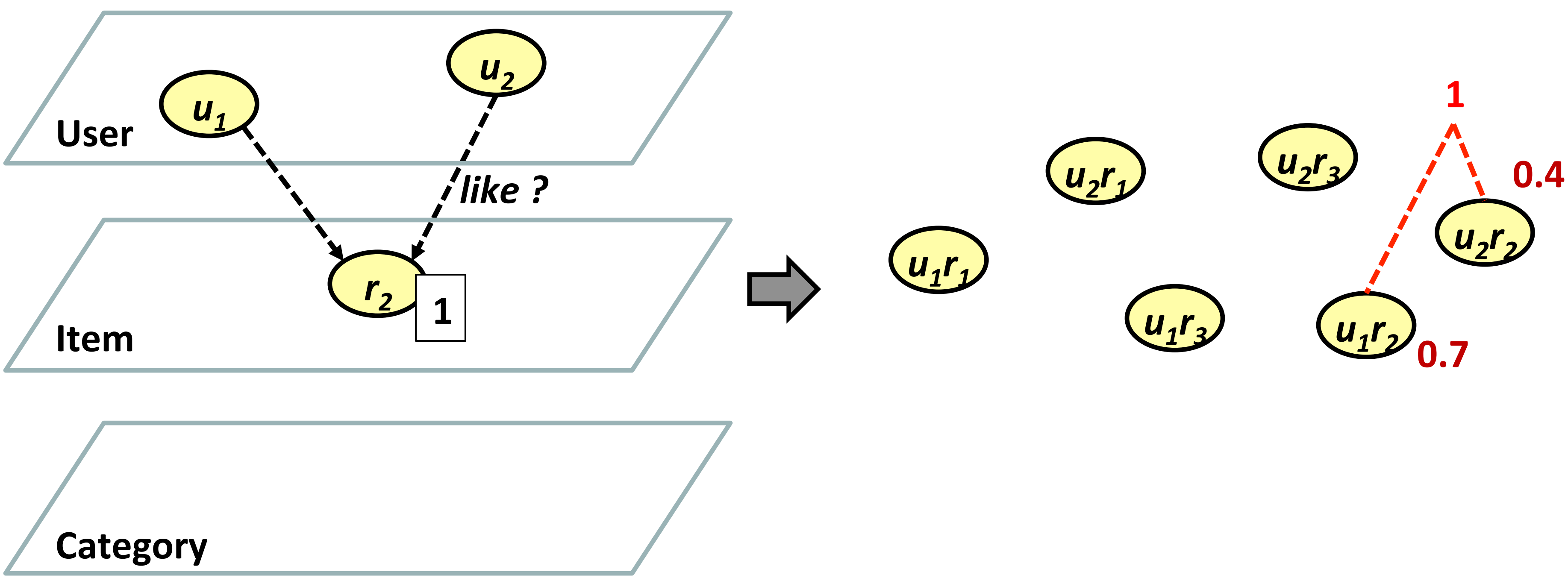
Intuition 3: Complex Heuristics (cont.)

- Similarly, we may have many complex hidden heuristics
  - Ex. **C2**: people tend to like items in same category of their owned items
  - That is, if u_1 like an item r_1, he/she may also like r_3 (in same c_1)



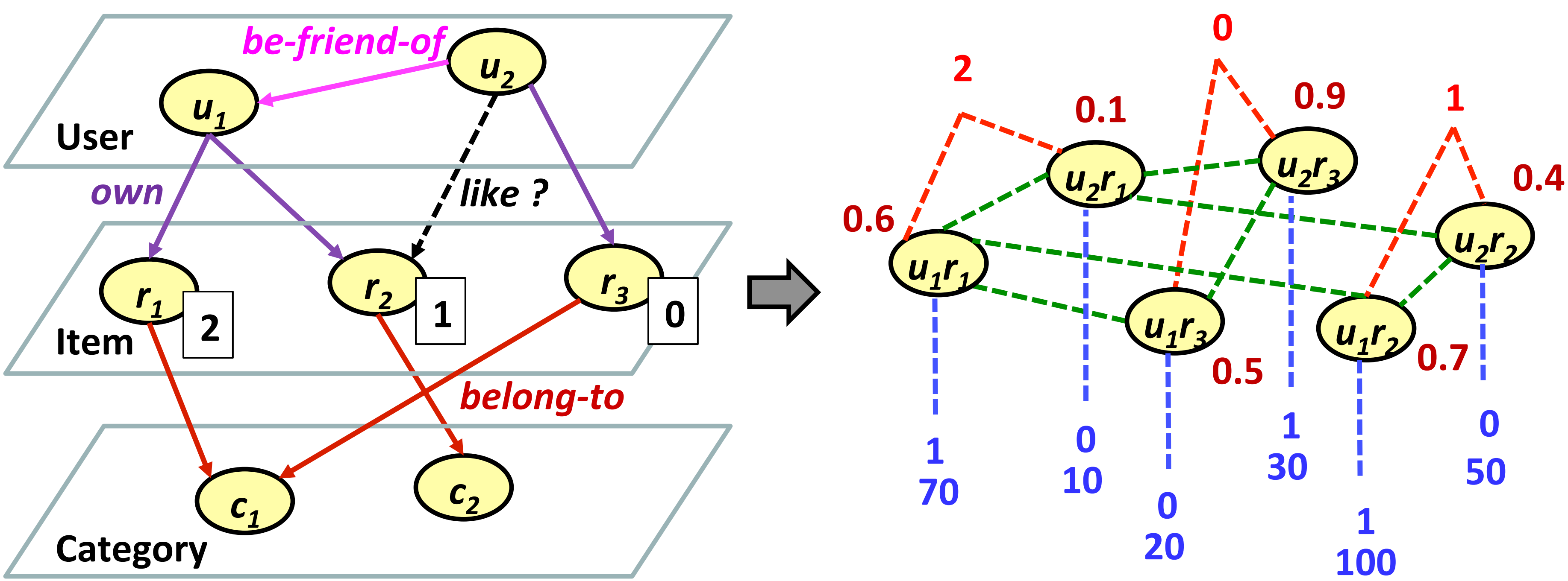
Intuition 4: Constraint Exists

- We know the total amount of 'like' for each item
  - We want the aggregative statistics of our predictions to match the known statistics
  - Ex. **N1**: assume predicted prob. $P(u_1 r_2) = 0.7$, $P(u_2 r_2) = 0.4$
  - We should predict $P(u_1 r_2) + P(u_2 r_2)$ as close to 1 as possible



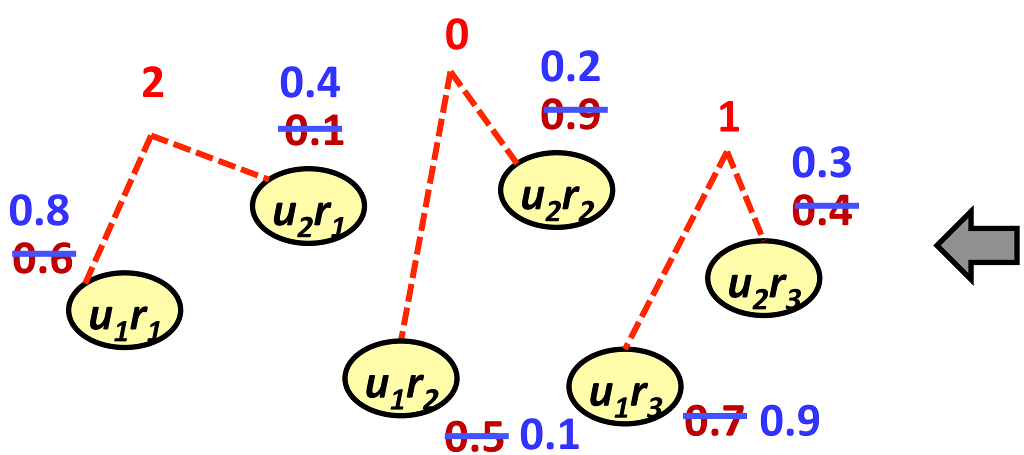
Intuition 5: Combining Heuristics

- Now we have many hypotheses, for instance
  - Characteristics of candidate pairs: **S1**, **S2**
  - Relations of candidate pairs: **C1**, **C2**
  - Constraint of candidate pairs : **N1**
- How do we know the importance (i.e. weights) of them?
- We modify a graphical model to learn weights and infer results



Intuition 6: Tuning Weights

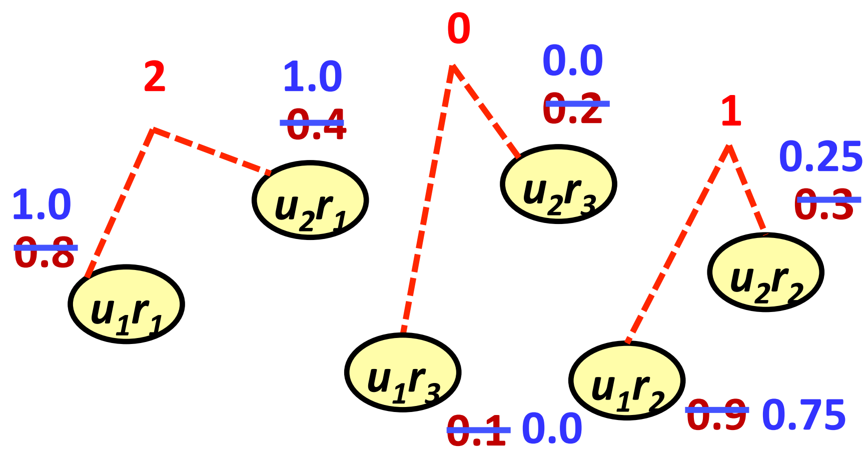
- In the graphical model, we have a weight for each heuristic, represented using potential functions
  - Ex. w_{S1}, w_{S2}, w_{C1}, w_{C2}, w_{N1}
  - The 5 weights are correlated to pairs or relations between pairs
- How can we tune the weights without labeled data ?
  - We can exploit aggregative statistics as guidance
  - For r_1, the two predictions (from u_1 and u_2) should be higher
  - For r_2, the two predictions should be → 0
  - For r_3, one prediction should be → 1, and another should be → 0



w_{S1} = ~~0.20~~	~~0.25~~	~~0.28~~	0.27
w_{S2} = ~~0.20~~	~~0.10~~	~~0.13~~	0.14
w_{C1} = ~~0.20~~	~~0.25~~	~~0.22~~	0.24
w_{C2} = ~~0.20~~	~~0.25~~	~~0.26~~	0.22
w_{N1} = ~~0.20~~	~~0.15~~	~~0.21~~	0.23

Intuition 7: Final Prediction

- After the weights are learned, we can predict final results
  - We can adjust probabilities directly to match aggregative statistics
  - For r_1 and r_3, we can adjust probabilities directly (special cases)
  - For r_2, we adjust probabilities to $P(u_1r_2) = 0.75$ and $P(u_2r_2) = 0.25$
- In real-world application, we need computational methods
  - To deal with large-scale datasets

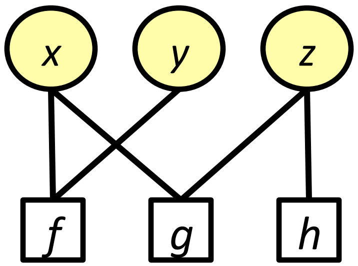


Challenges

- How can we learn from not only the unlabeled data we have, but also incorporate the abovementioned knowledge into the framework?
  - Furthermore, we want to avoid the consequence of the incorrect hypotheses
- How can learning be conducted without labeled data?

Factor Graph Model (FGM) is Exploited

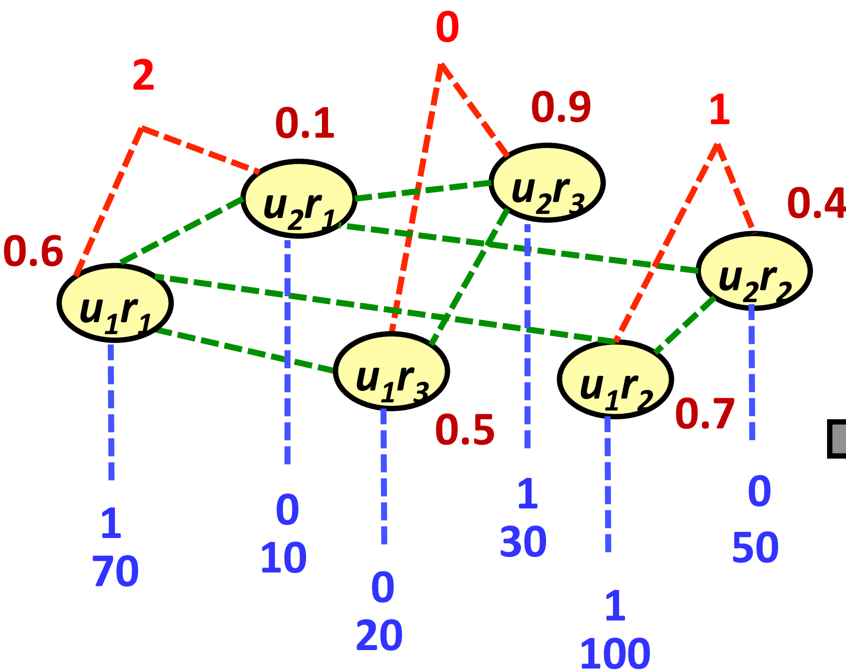
- Introduction to FGM
  - Deal with complex global functions with many variables
  - Split the joint distribution as a product of simpler local functions
  - Represent such factorization as a bipartite graph
  - Example
    - Let *x*, *y*, and *z* be random variables with different distributions
    - Maximize joint distribution $P(x, y, z)$
    - Suppose $P(x, y, z) = f(x, y)\, g(x, z)\, h(z)$
    - Infer *x*, *y*, *z* to maximize $P(x, y, z)$

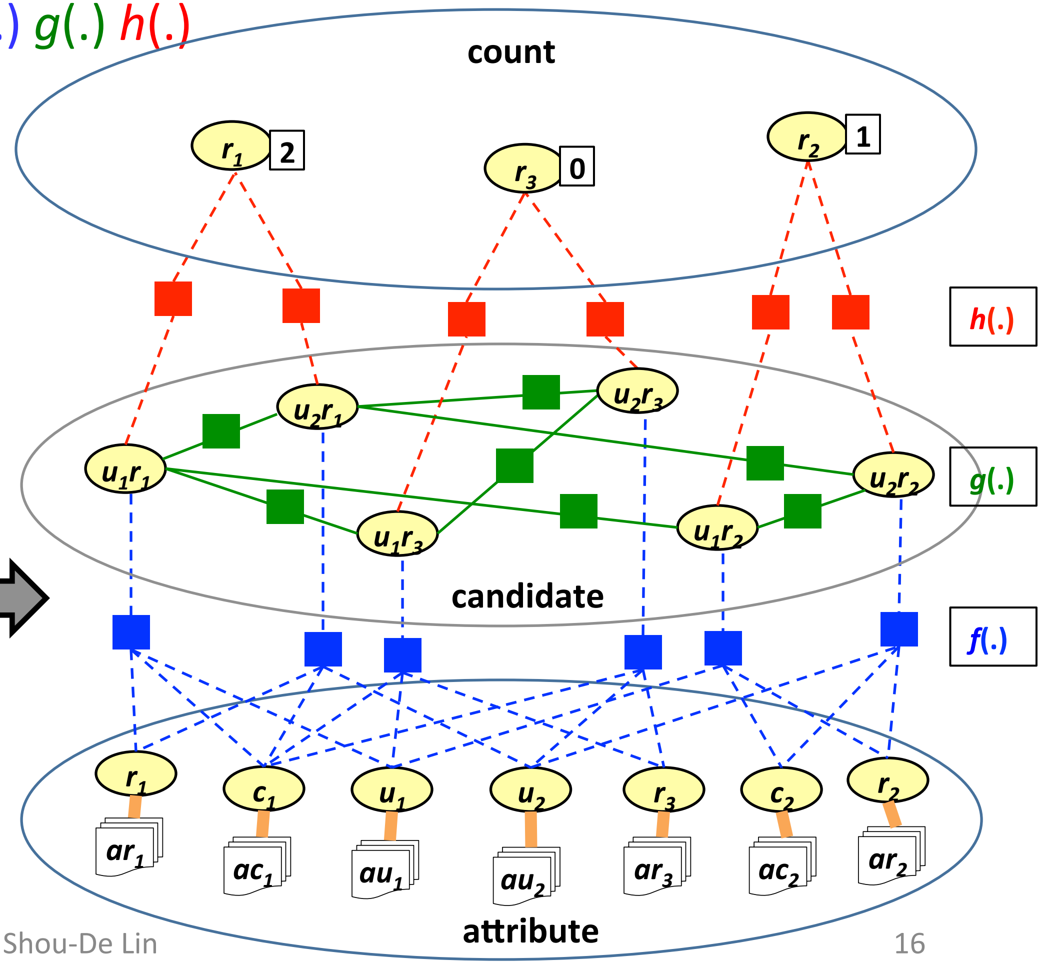


- Reasons to exploit FGM
  - Integrate attributes and predictions as random variables
  - Model knowledge or hypothesis as potential functions (and the weights can be learned)
  - Predict links using aggregative statistics via learning and inference

FGM with Aggregative Statistics (FGM-AS)

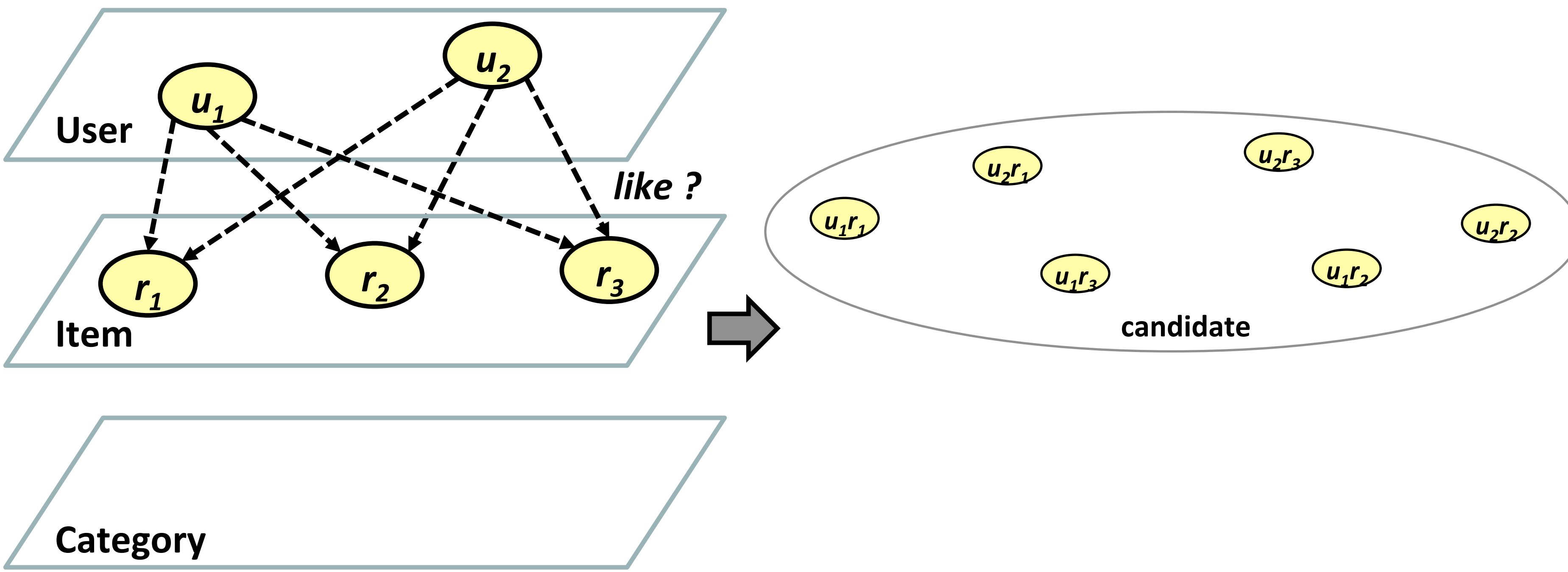
- **Random variables**: candidate, attribute, count
- **Potential functions**: $f(.)$, $g(.)$, $h(.)$
- **Learning**: adjust parameters in $f(.)$, $g(.)$, $h(.)$
- **Inference**: select $y = u_i r_j$ to max $\Pi\ f(.)\ g(.)\ h(.)$
- **Output**: positive marginal prob. of y





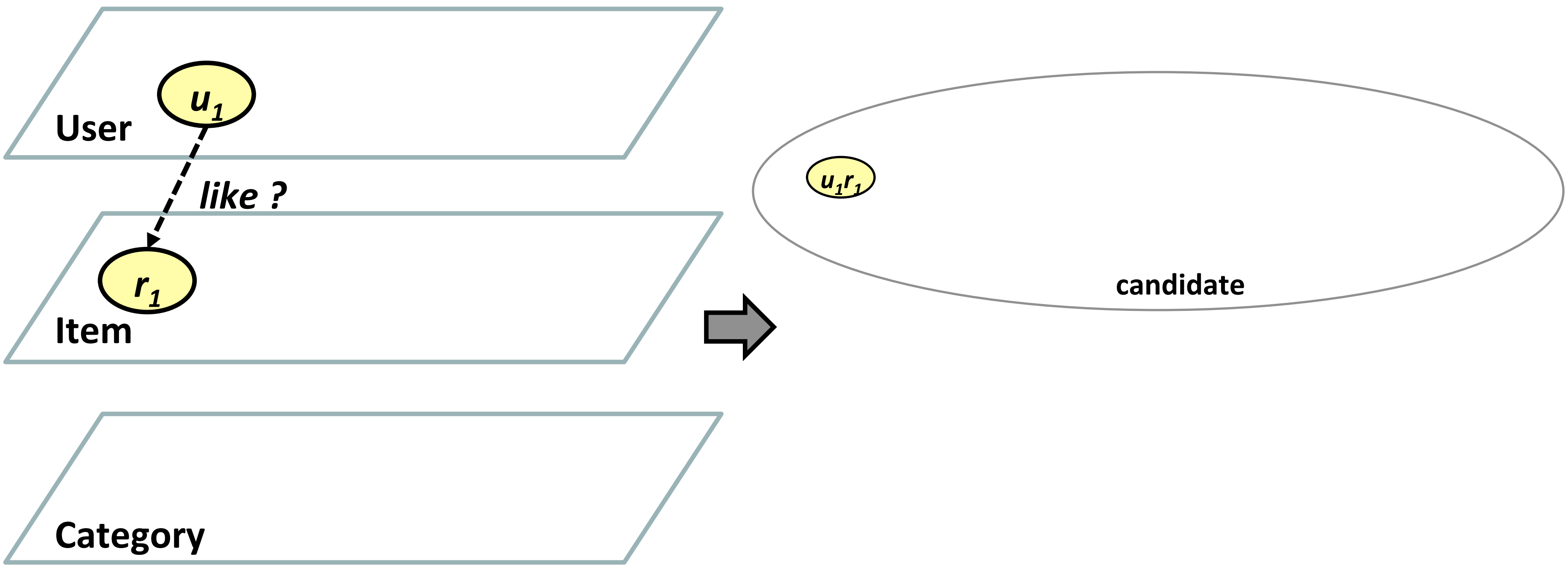
Candidate Variables

- Possible links to predicted as random variables
  - Let *y* = <*user*, *item*> pairs be candidate variables
  - Binary variable *y* = 1 if there is a link, otherwise 0
  - Thus, we want to infer the positive marginal prob. of *y*'s with FGM



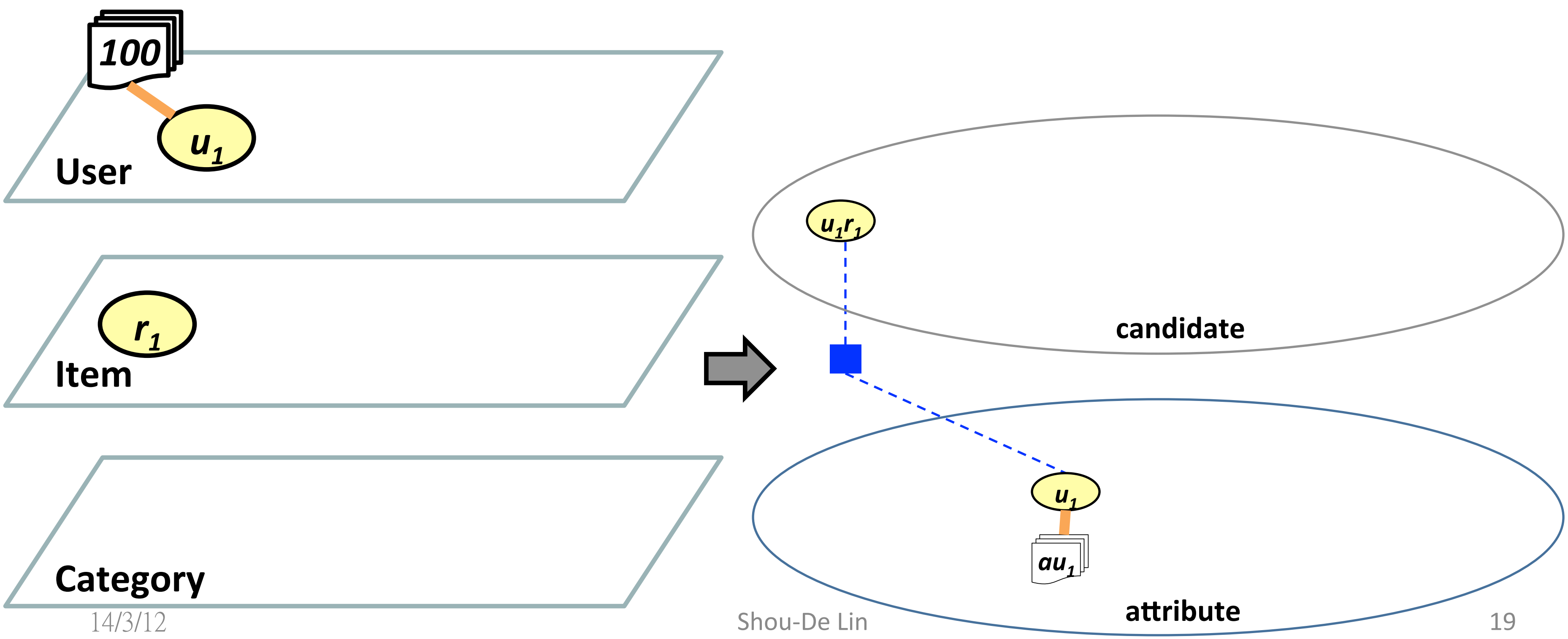
Attribute and *f*(.)

- Intuition 1: simple heuristics



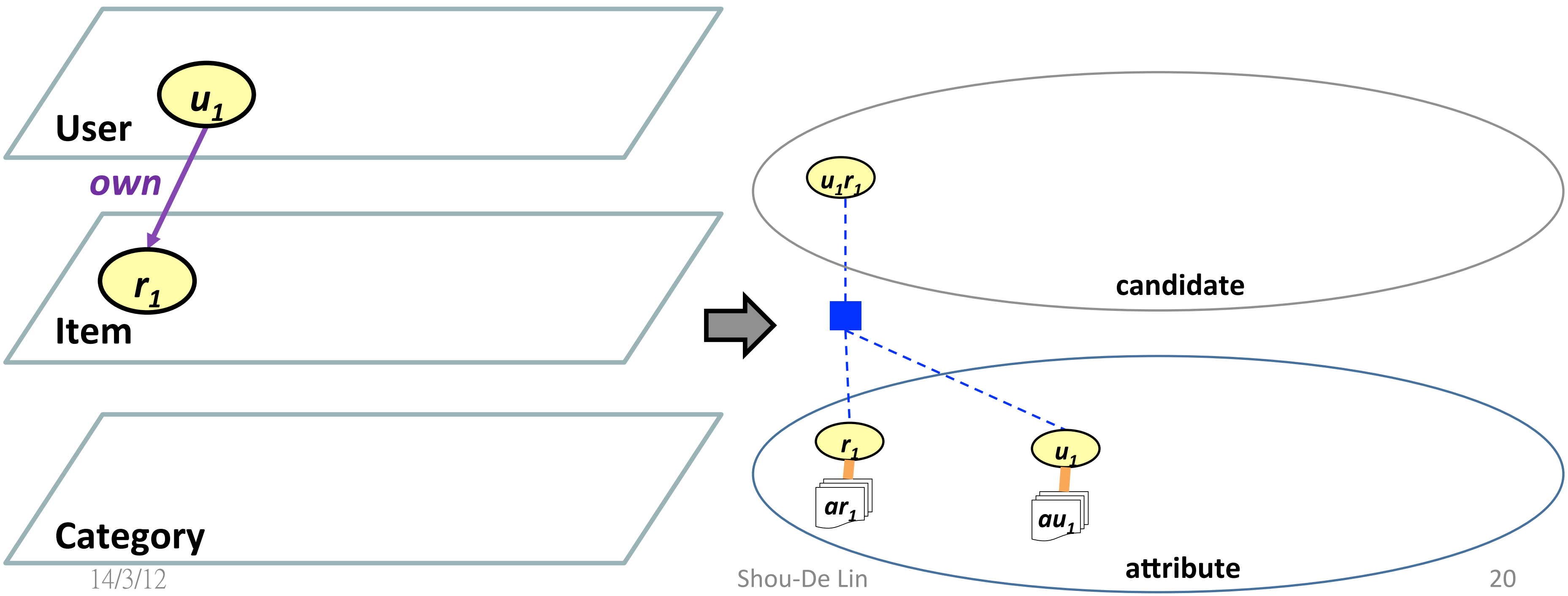
Attribute and *f*(.)

- Intuition 1: simple heuristics
  - User Friendship (UF) = # of friends of u_1= 100 (integer)
    - "People with more friends tend to like every items"



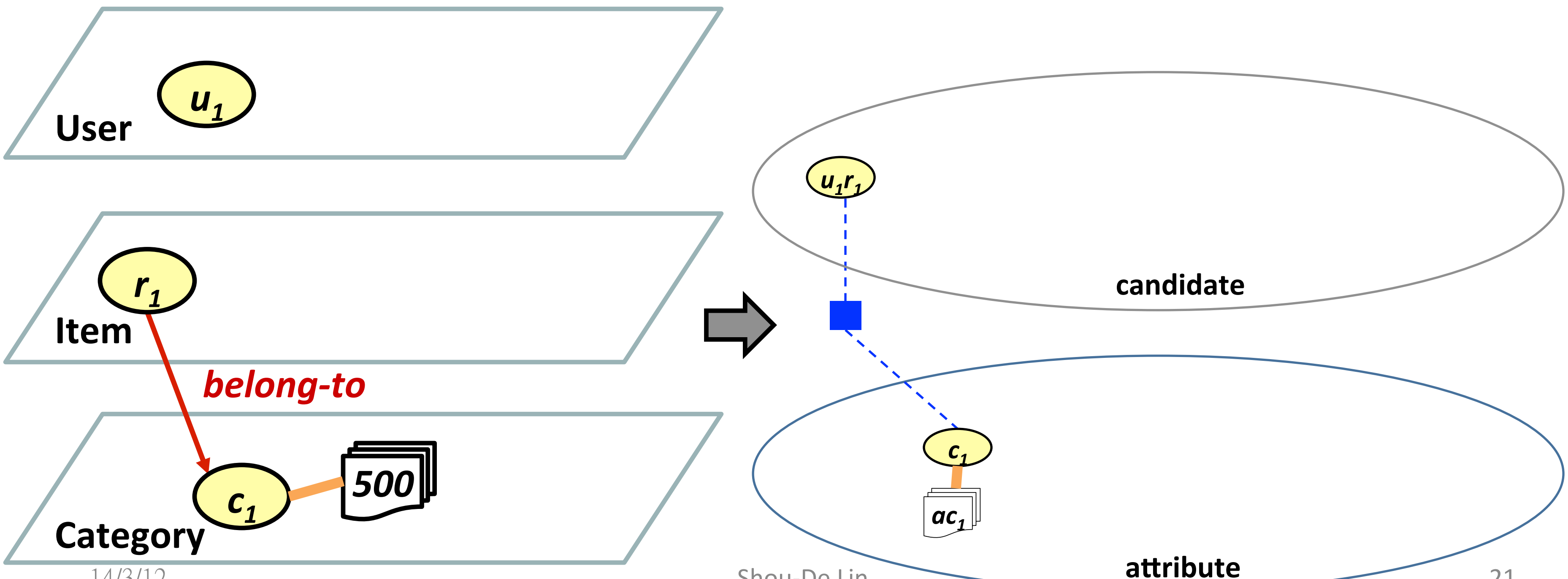
Attribute and *f*(.)

- Intuition 2: simple heuristics
  - User Friendship (UF)
  - Item Ownership (IO) = whether u_1 owns r_1 = 1(binary)
    - "People tend to like their own items"



Attribute and $f(.)$

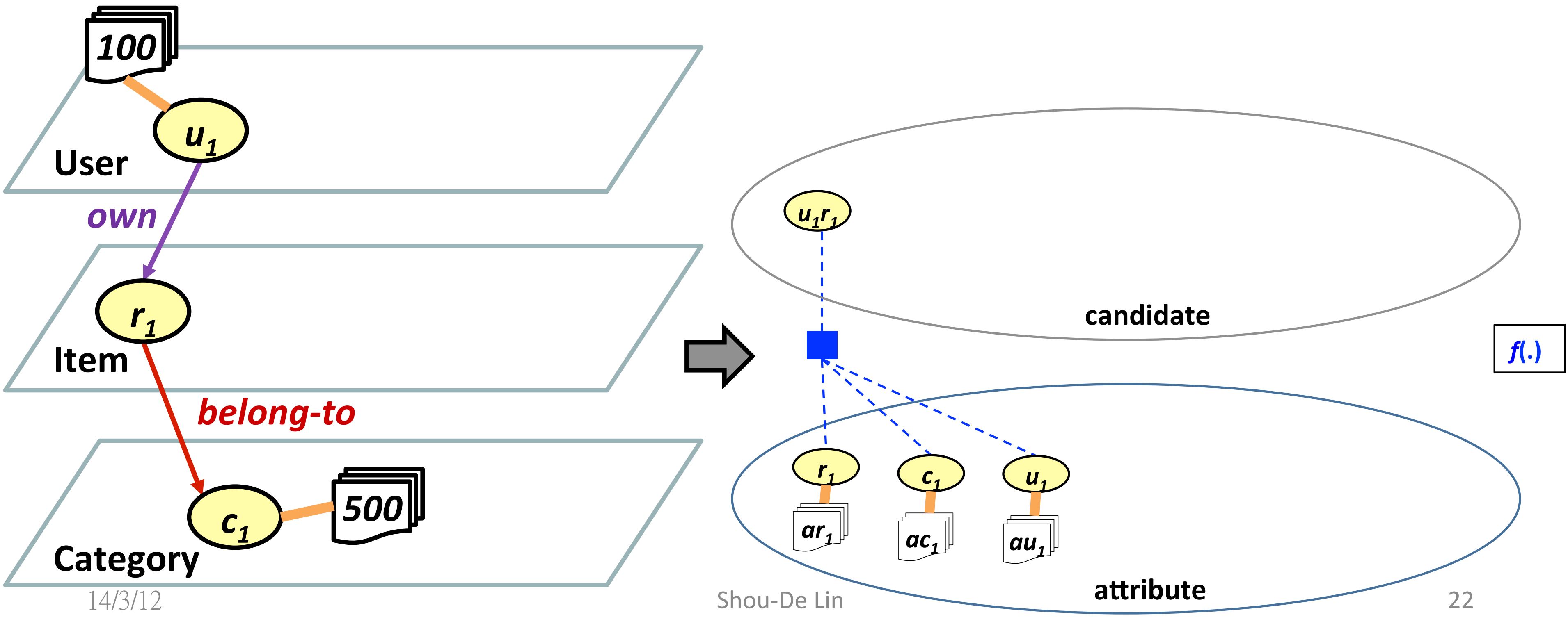
- Intuition 2: simple heuristics
  - User Friendship (UF)
  - Item Ownership (IO)
  - Category Popularity (CP) = # of items in c_1 = 500 (integer)
    - "People tend to like items in popular categories"



Attribute and *f*(.)

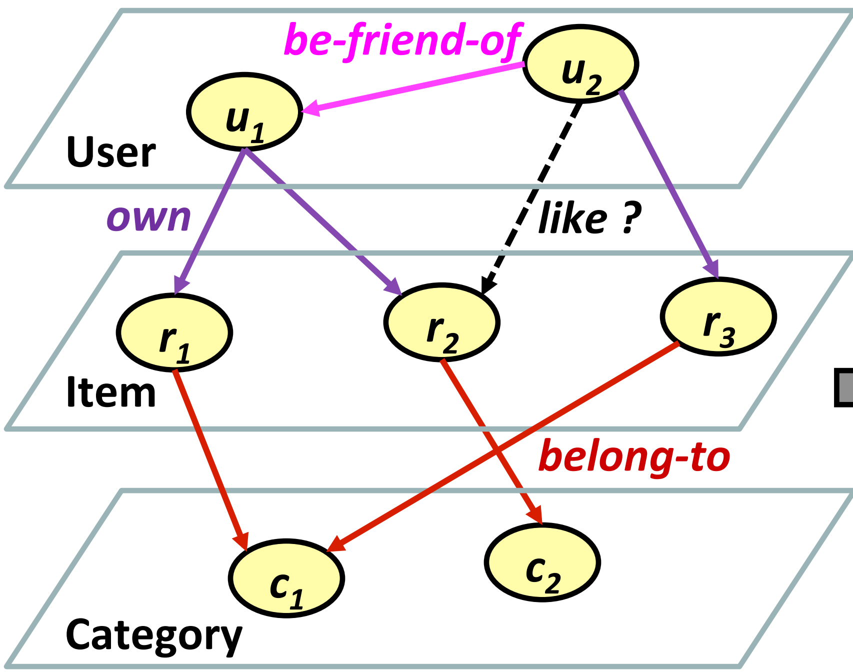
- Intuition 2: simple heuristics
  - User Friendship (UF)
  - Item Ownership (IO)
  - Category Popularity (CP)
  - *f*(.) is linear exponential combination of UF, IO and CP

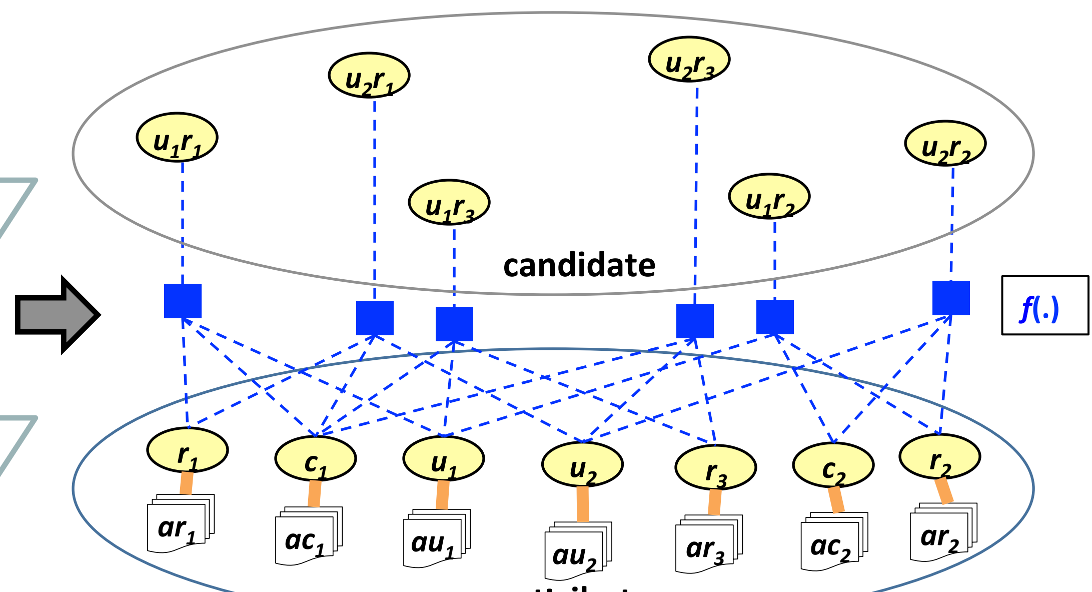
$$f(y) = \frac{1}{Z_\alpha} \exp\{\alpha \cdot \, < UF, IO, CP >\}$$
$$= \frac{1}{Z_\alpha} \exp\{\alpha \cdot f'(y)\}$$



Attribute and *f*(.)

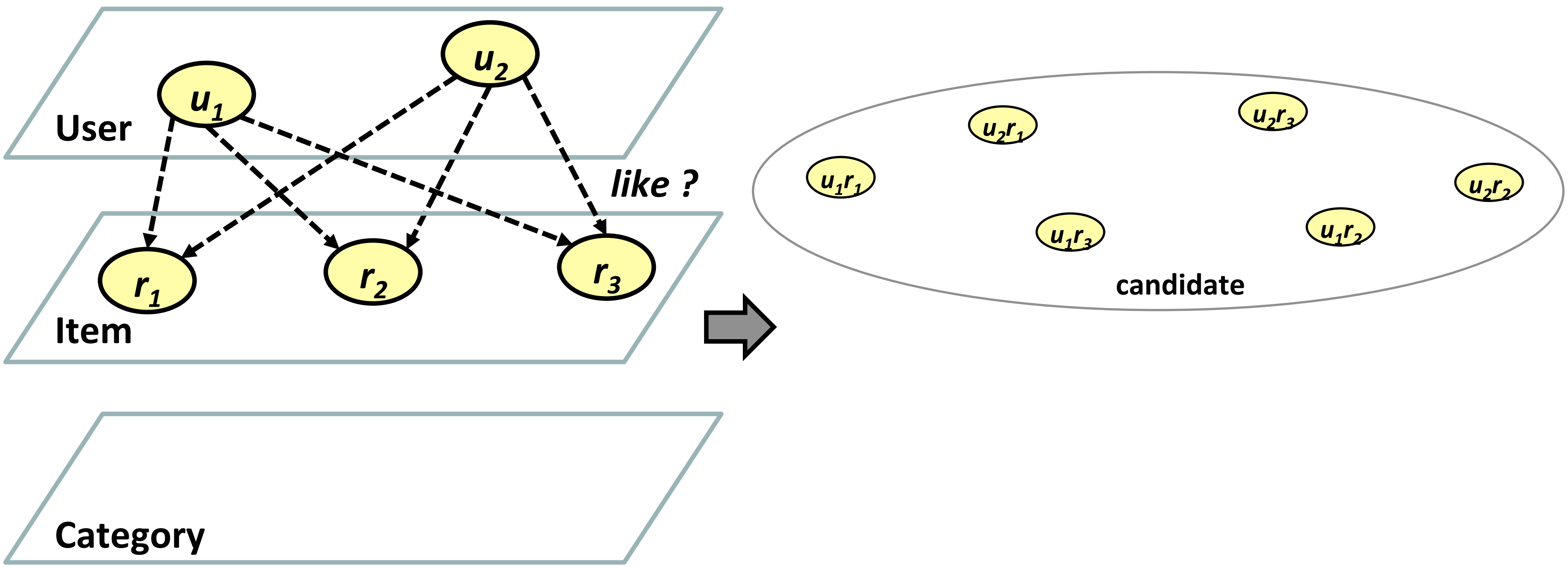
- Intuition 2: simple heuristics
  - All *f*(.) can be constructed in the similar way
  - Each candidate pair has a corresponding *f*(.)





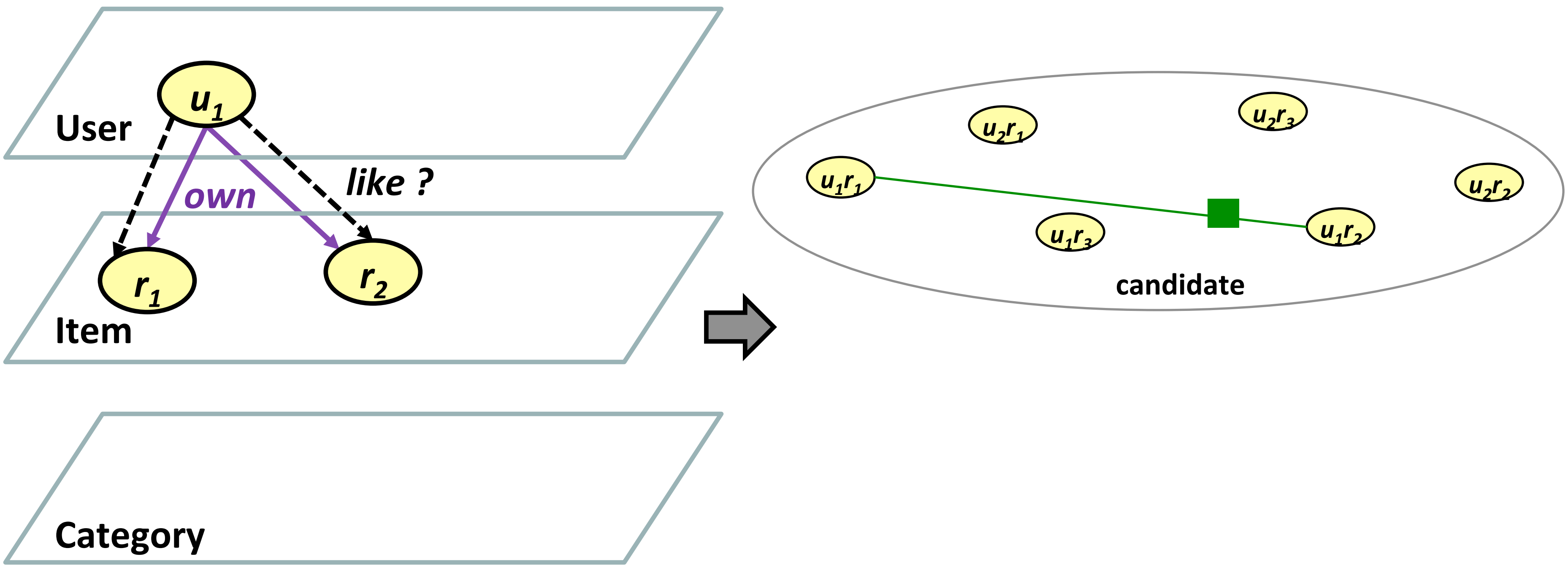
$g(.)$

- Intuition 3: complex heuristics



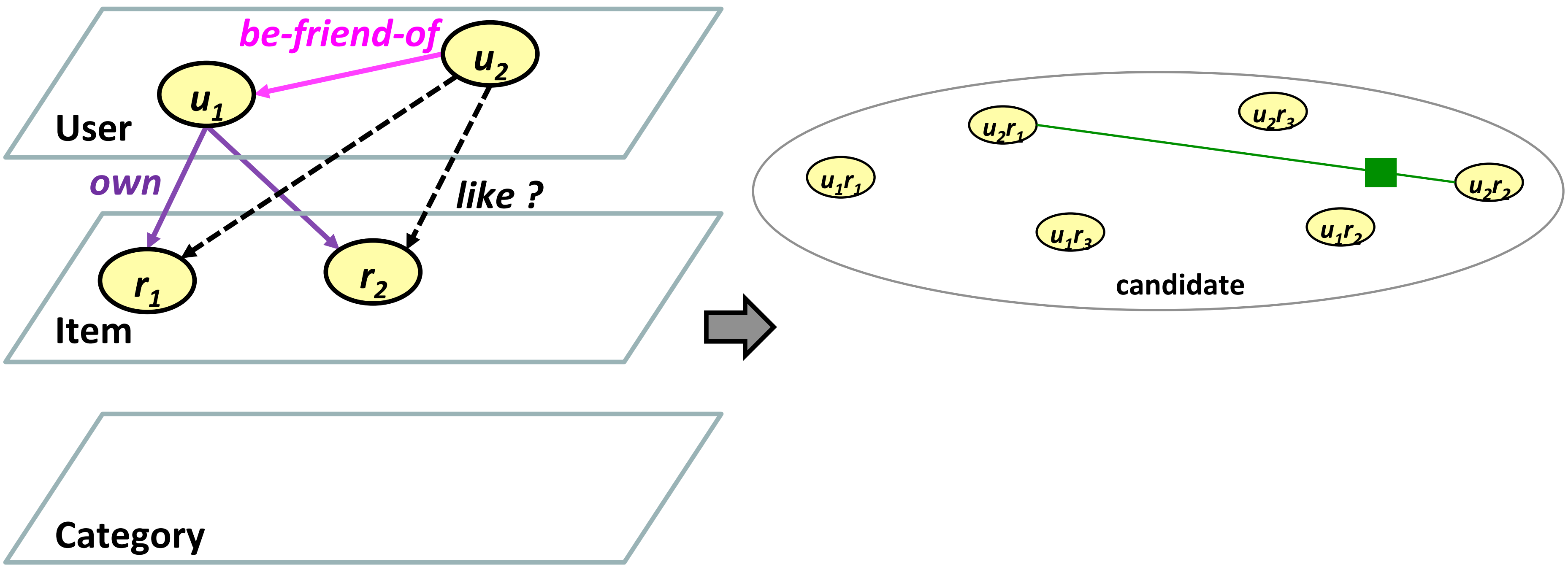
g(.)

- Intuition 3: complex heuristics
  - Owner-Identification (OI) = u_1 likes their owned post (r_1 , r_2) = 1
    - "People tend to like their owned items in similar extend" (binary)



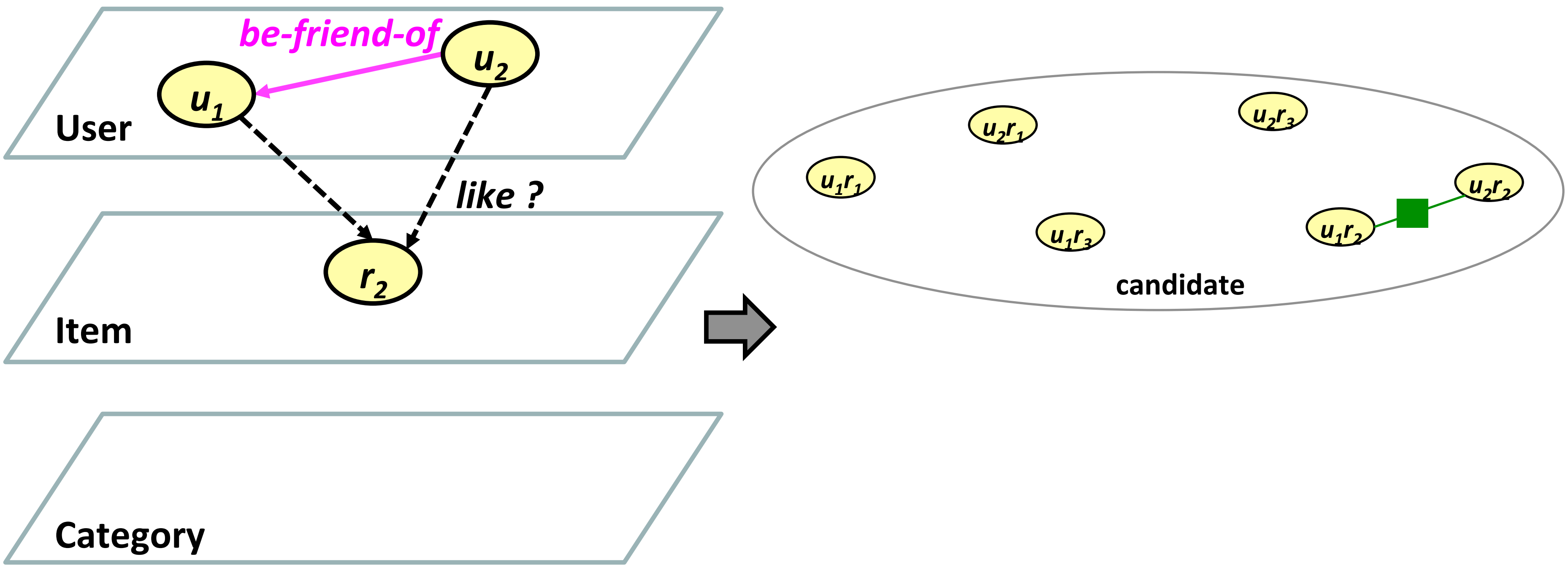
$g(.)$

- Intuition 3: complex heuristics
  - Owner-Identification (OI)
  - Friend-Identification (FI) = u_2 likes both u_1's post $(r_1, r_2) = 1$
    - "People tend to like friends' items in similar extend" (binary)



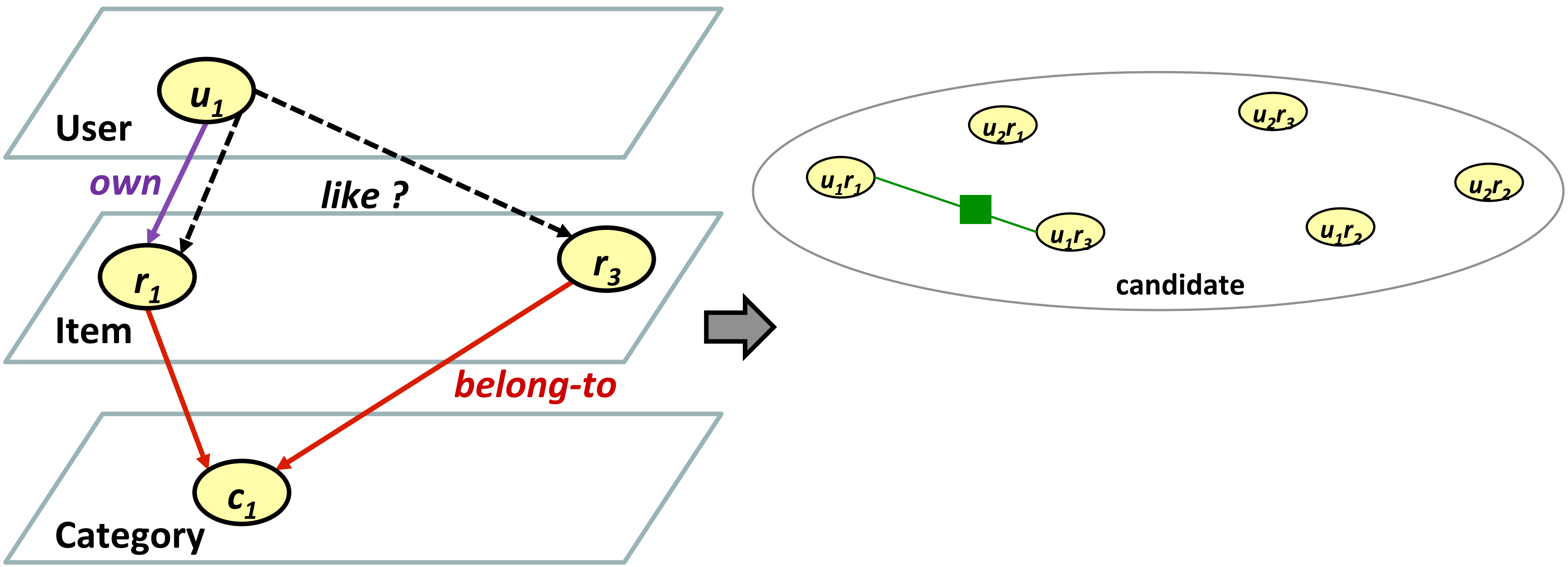
$g(.)$

- Intuition 3: complex heuristics
  - Friend-Identification (FI)
    - "People tend to have similar tastes as their friends" (binary)



$g(.)$

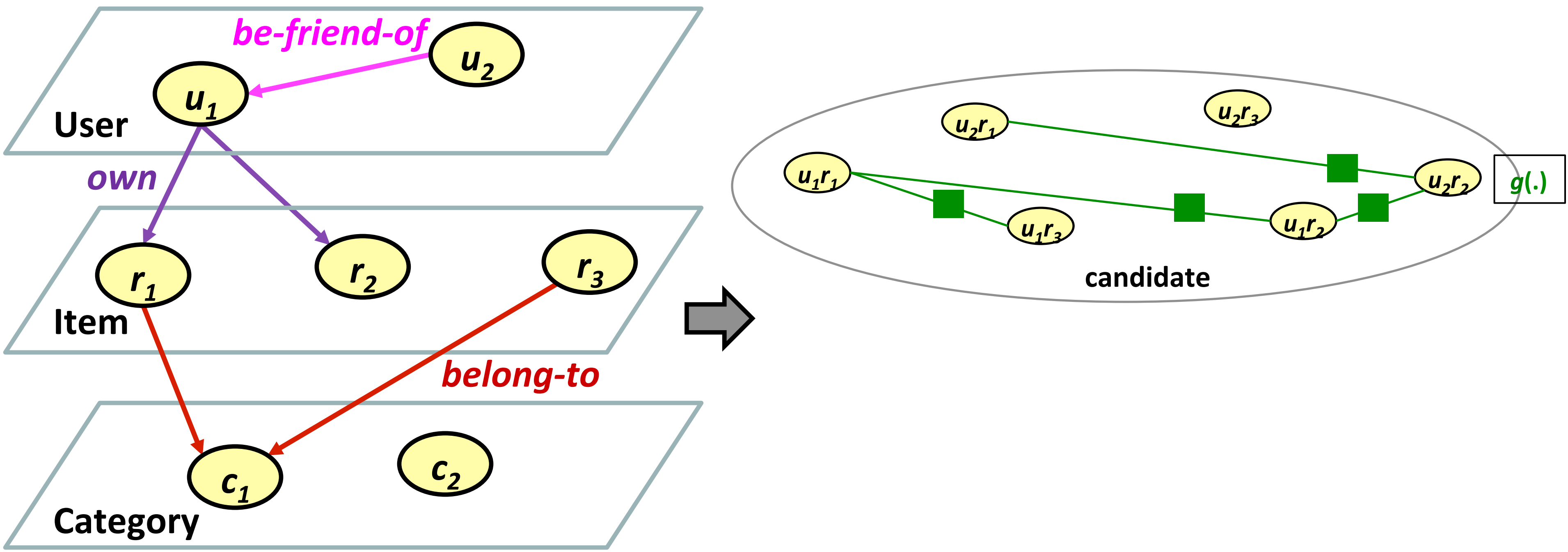
- Intuition 3: complex heuristics
  - Owner-Identification (OI)
  - Friend-Identification (FI)
  - Owner-Friend (OF)
  - Co-category (CC) = u_1 like r_3 as u_1 like r_1 = 1
    - "People tend to like items in the same category of their own items" (binary)



g(.)

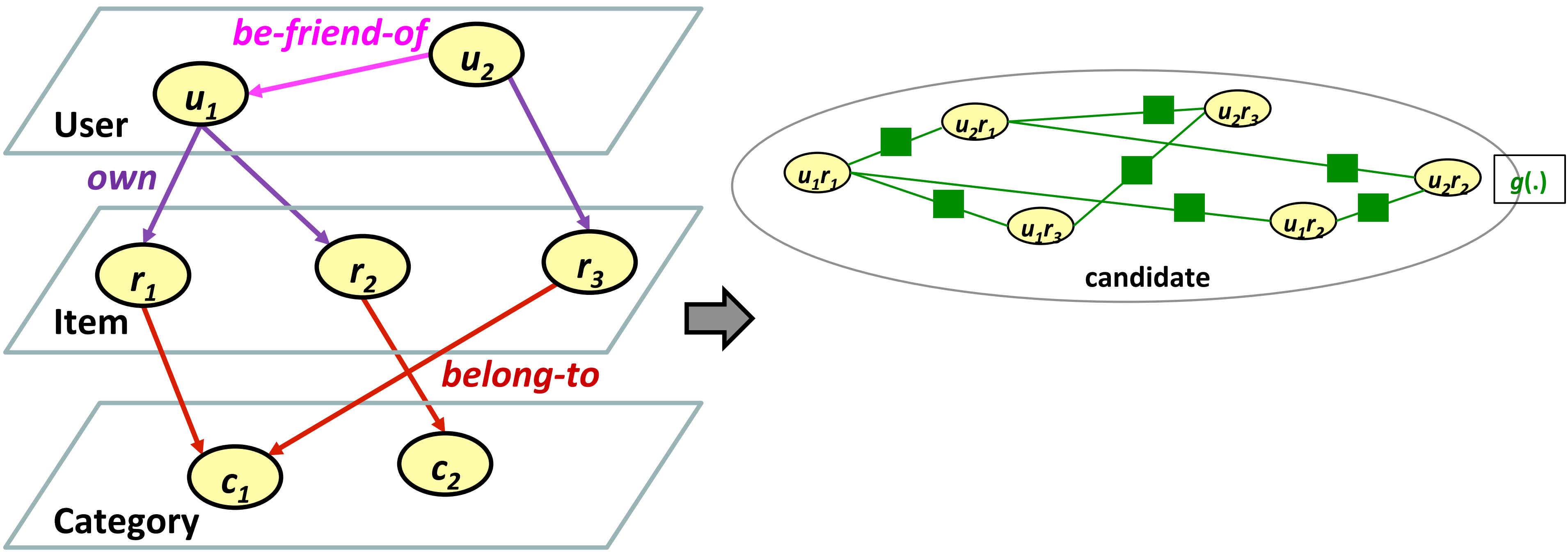
- Intuition 3: complex heuristics
  - Owner-Identification (OI)
  - Friend-Identification (FI)
  - Owner-Friend (OF)
  - Co-category (CC)
  - g(.) is linear exponential combination of OI, FI, OF and CC

$$g(y) = \frac{1}{Z_\beta} \exp\{\beta \cdot < OI, FI, OF, CC >\}$$
$$= \frac{1}{Z_\beta} \exp\{\beta \cdot g'(y)\}$$



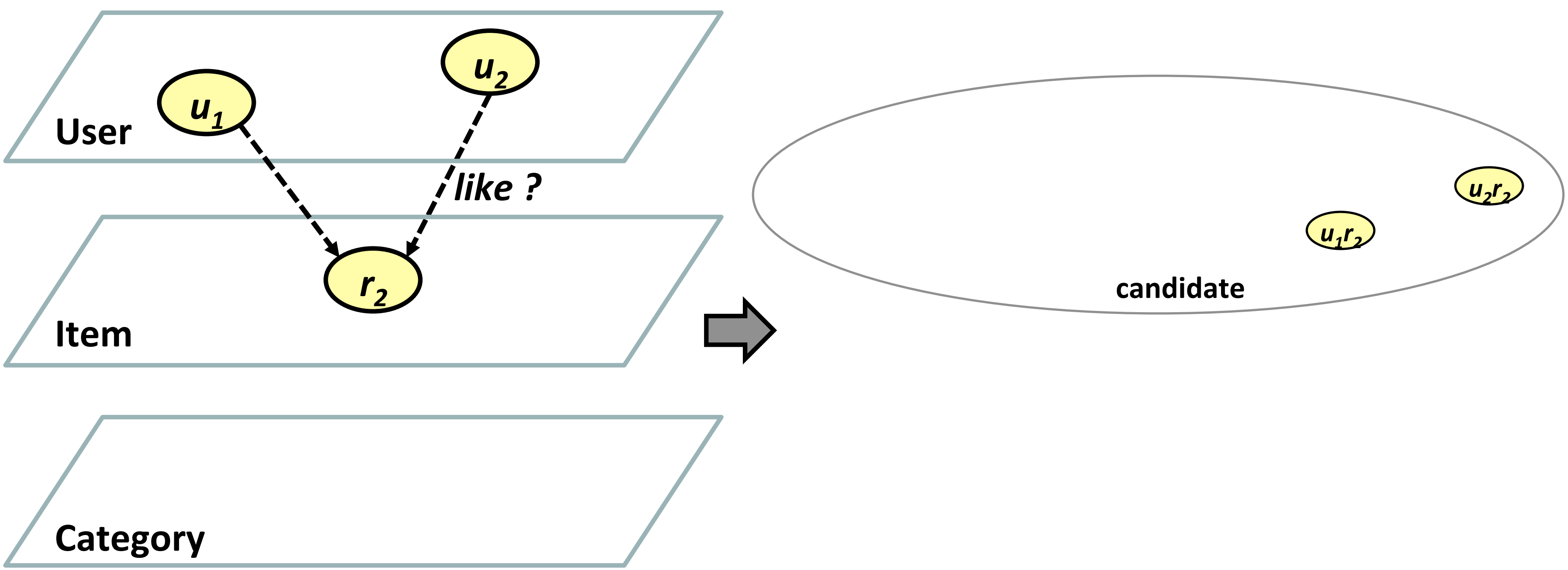
g(.)

- Intuition 3: complex heuristics
  - All g(.) can be constructed in the similar way
  - If g(.) = 0, we simple ignore the link



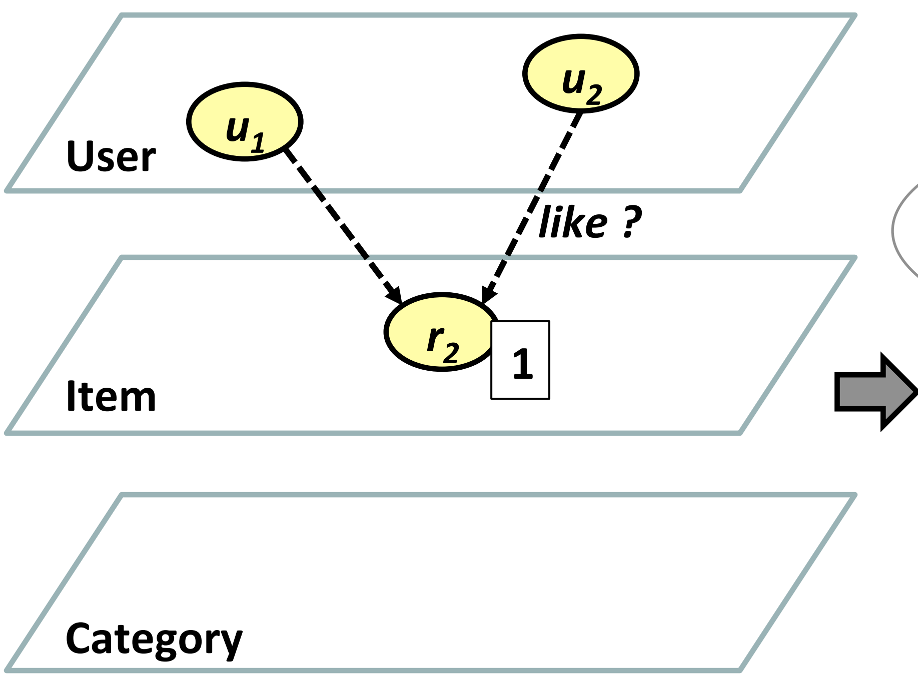
Count and *h*(.)

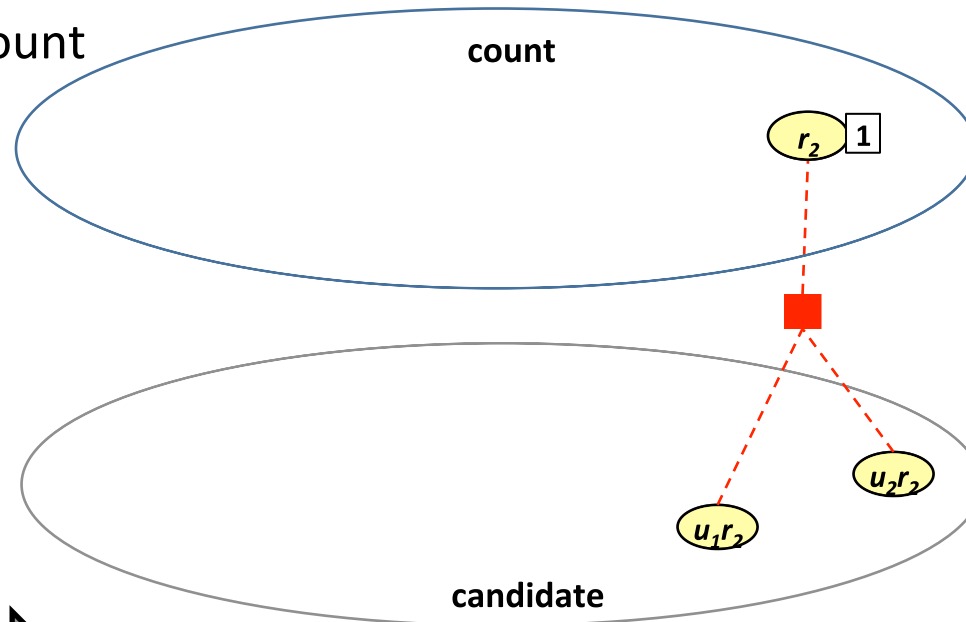
- Intuition 4: constraint heuristics



Count and *h*(.)

- Intuition 4: constraint heuristics
  - Candidate-Count (CT) = the closeness of the following two terms
    - *ΣP* = sum of predicted probabilities of "*like*" to an item
    - *t*(*y*) = observed total "*like*" count

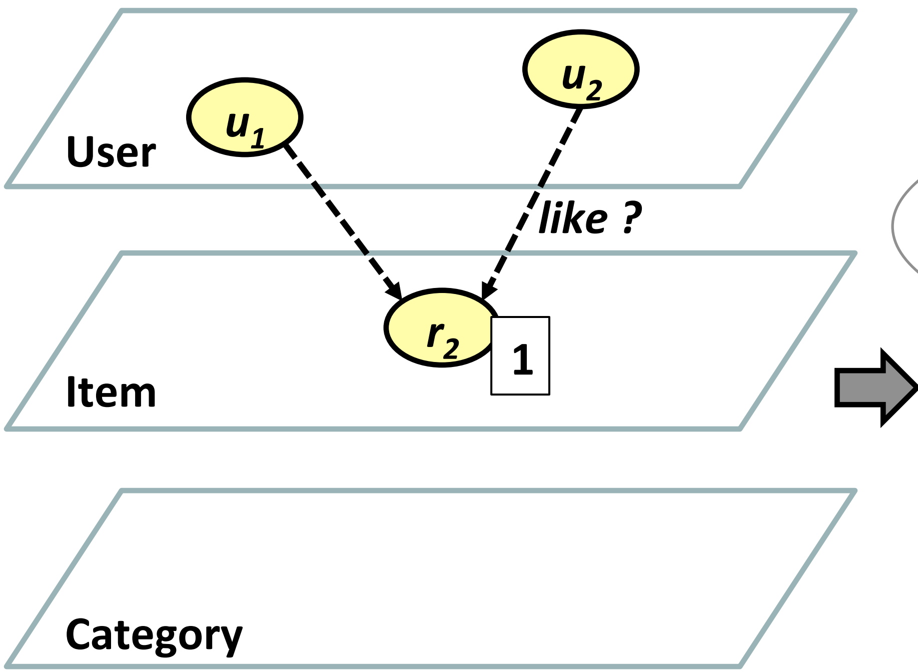


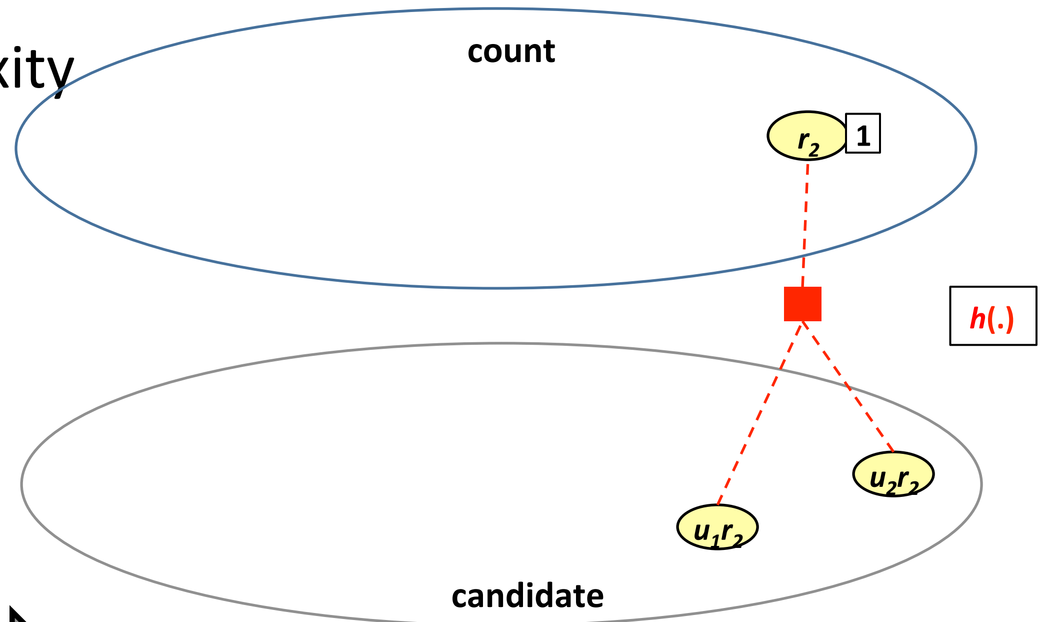


$$CT(y_i) = 1 - \left| \frac{t(y_i) - \sum_{y_j \in Y, r(y_j) = r(y_i)} P(y_j = 1)}{|user|} \right|$$

Count and *h*(.)

- Intuition 4: constraint heuristics
  - Candidate-Count (CT)
  - *h*(.) is linear exponential combination of CT
  - Split *h*(.) to reduce complexity



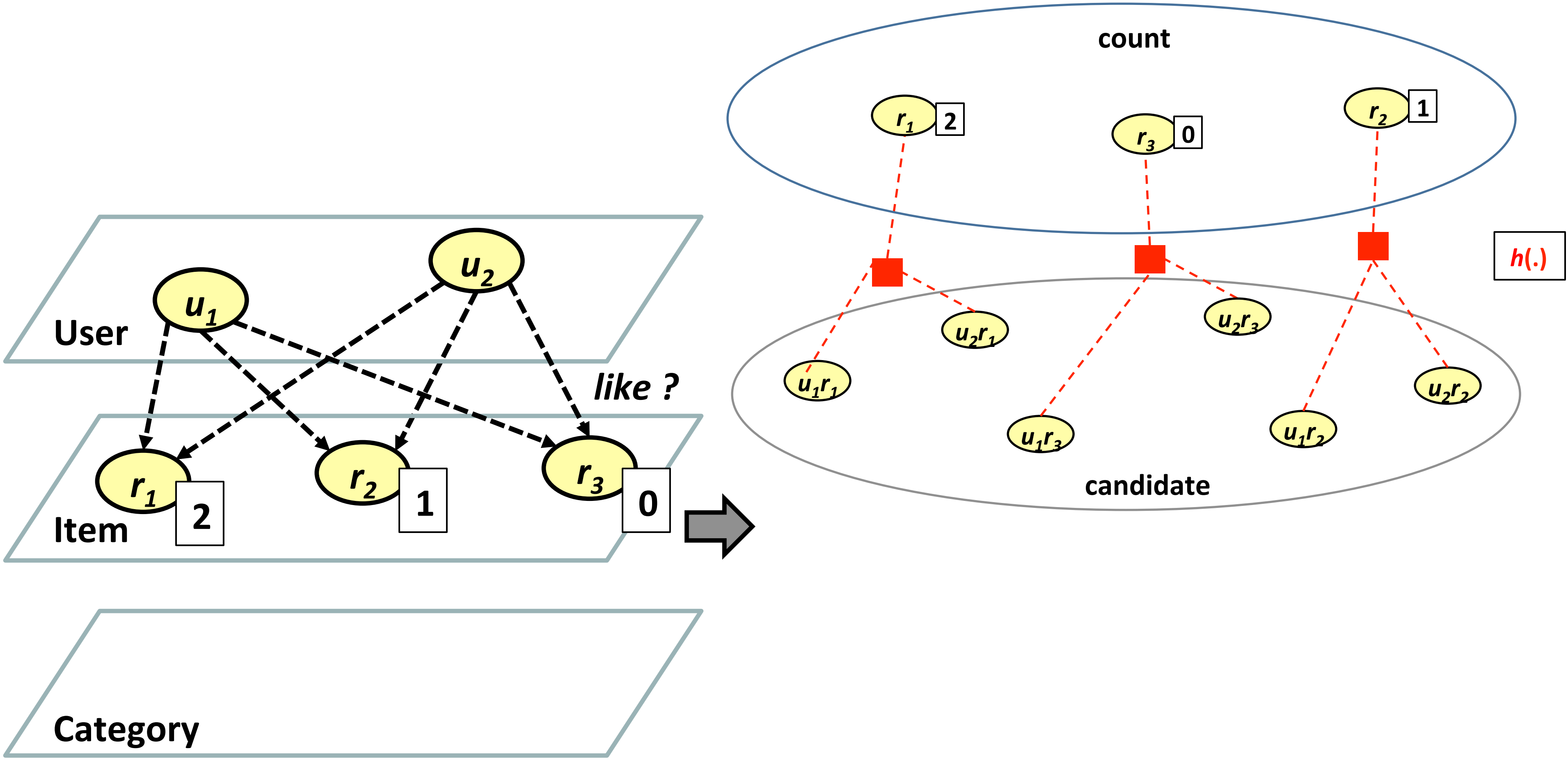


$$h(y) = \frac{1}{Z_\gamma}\exp\{\gamma \cdot < CT >\}$$

$$= \frac{1}{Z_\gamma}\exp\{\gamma \cdot h'(y)\}$$

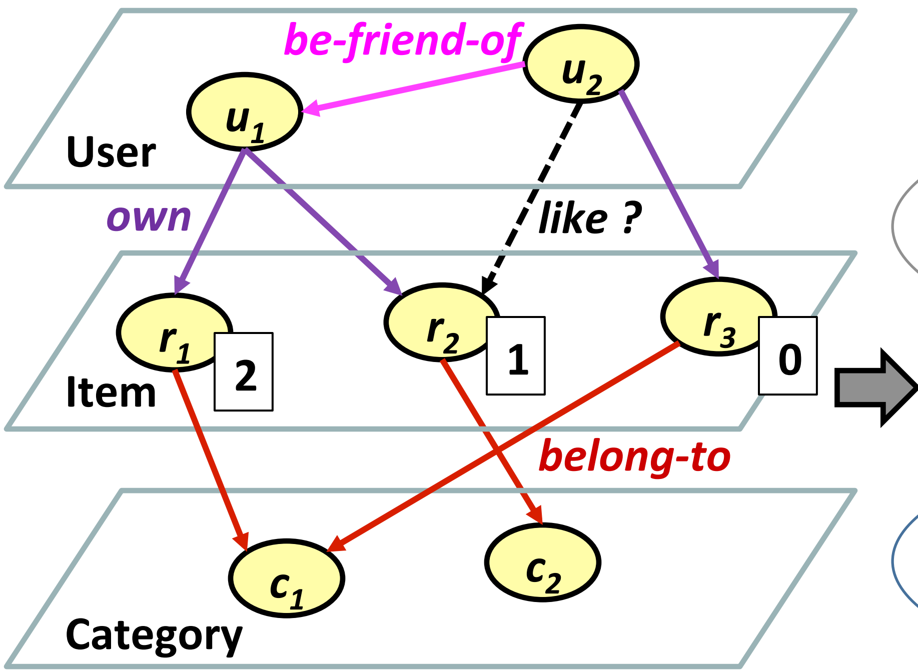
Count and *h*(.)

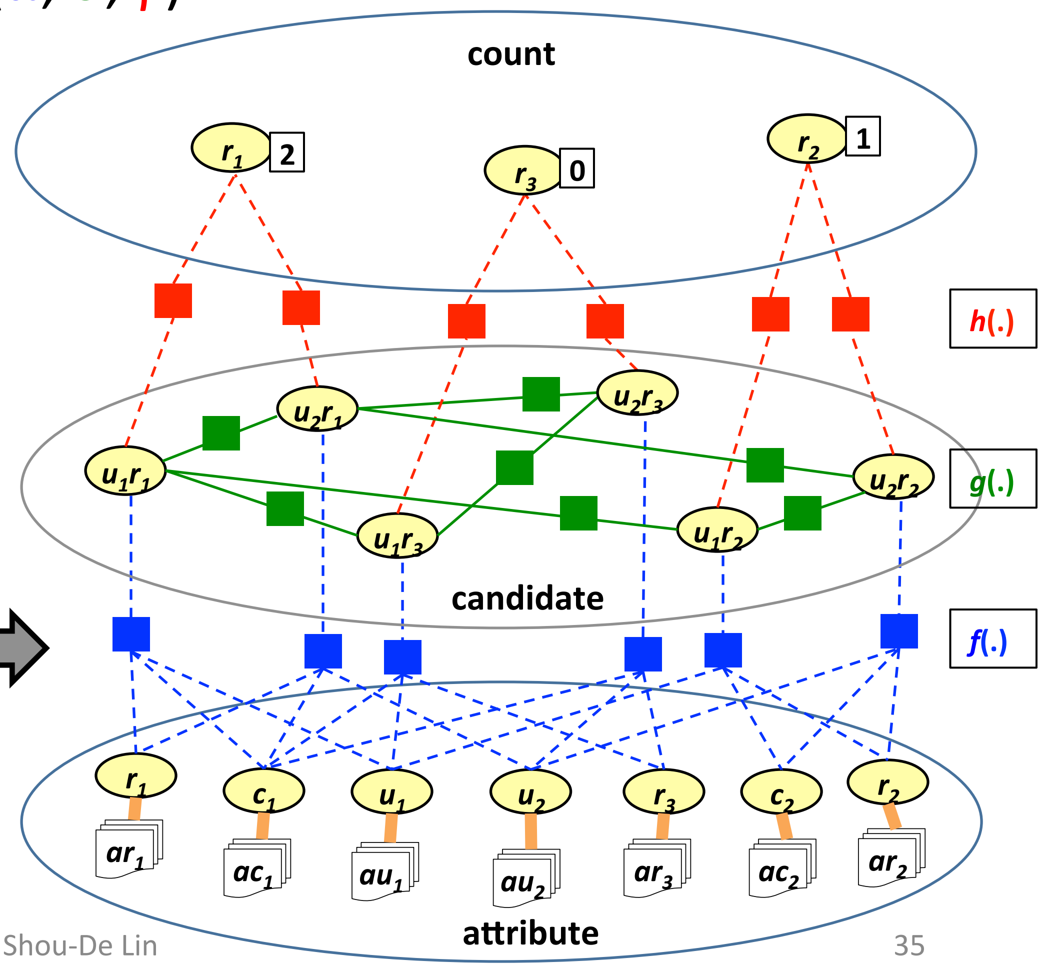
- Intuition 4: constraint heuristics
  - All *h*(.) can be constructed in the similar way



Count and *h*(.)

- Intuition 5: combining heuristics
  - Joint distribution $P(.) = \Pi\, f(.)\, g(.)\, h(.)$
  - Weighting parameters $\theta = (\alpha, \beta, \gamma)$
  - Let $S = \Sigma_y f'(y)\, g'(y)\, h'(y)$



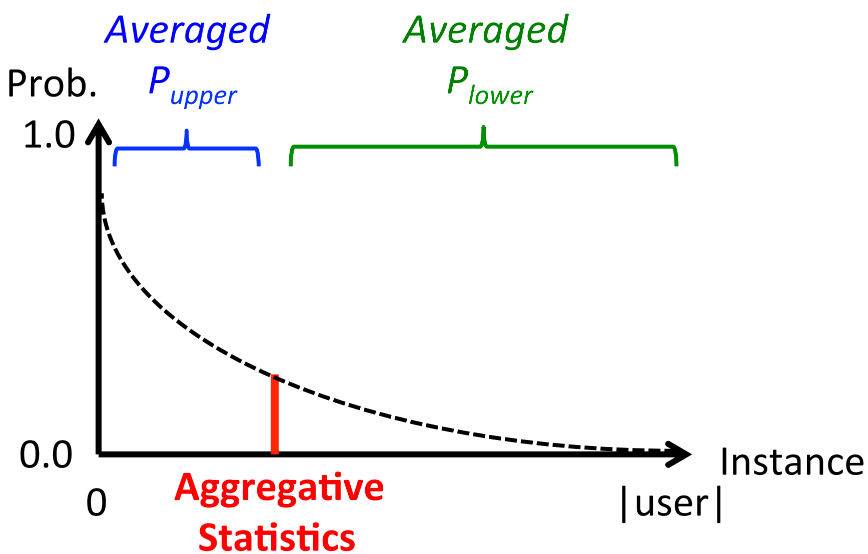


Ranked-Margin Learning

- How can we learn without labels?
  - We want to adjust weighting parameters θ so that
    - 'suspicious' users have high probabilities to "*like*"
    - The rest users have low probabilities to "*like*"
  - Thus, we want to maximize

$$\frac{\partial(\theta, r)}{\partial \theta} = \mathbb{E}_{P_{\theta, upper}} S - \mathbb{E}_{P_{\theta, lower}} S$$

    - $Diff_{margin}$ = Averaged $P_{\vartheta, upper}$ – Averaged $P_{\vartheta, lower}$
- Learning can be done similar to SGD
  - Repeat
    - Run inference algorithm
    - For each item
      - Compute gradient
      - Update parameter
    - End
  - Until convergence



Two-Stage Inference

$$CT(y_i) = 1 - \left| \frac{t(y_i) - \sum_{y_j \in Y, r(y_j)=r(y_i)} P(y_j = 1)}{|user|} \right|$$

- Intuition 7: final prediction
  - After learning parameters, we do inference for final prediction
  - link probability = marginal probability of *y*
- To compute CT in *h*(.), the term *ΣP* is required
  - Note that *ΣP* is not a random variable
  - We split *h*(.), thus *ΣP* need to be individually computed
  - Thus, conventional inference cannot be applied directly
- Therefore, we design an two-stage inference algorithm
  - **Stage 1**: infer using *f*(.), *g*(.) only (set all *h*(.) = 1) to get *ΣP*
  - **Stage 2**: compute *h*(.) using *ΣP*, then infer using *f*(.), *g*(.), *h*(.)

Scenario and Dataset

- We study 4 scenarios using real-world datasets
  - Preference prediction (Foursquare)
  - Repost prediction (Twitter)
  - Response prediction (Plurk)
  - Citation prediction (DBLP)

Random Variable		**Foursquare**	**Twitter**	**Plurk**	**DBLP**
Candidate	y	Like	Retweet	Response	Citation
Attribute	u	User	User	User	User
	r	Tip	Tweet	Message	Paper
	c	Venue	Term	Topic	Keyword
Count	t	Likes per tip	Retweets per tweet	Responses per message	Citations per paper

Statistics of Dataset

- We hide unseen links as ground truth for evaluation
  - Foursquare: still very few preferences are revealed
- Unseen-type links are sparse comparing to all candidates
  - Foursquare: |unseen| / (|user| * |item|) = $1.22 * 10^{-6}$

Property		Foursquare	Twitter	Plurk	DBLP
Node	User	71,634	69,026	190,853	102,304
	Item	180,684	55,375	352,376	221,935
	Category	16,961	100	100	100
	Total	**269,279**	**124,501**	**543,329**	**324,339**
Link	Be-friend-of	724,378	21,979,021	2,151,351	245,391
	Own	180,684	55,375	352,376	221,935
	Belong-to	180,684	55,375	352,376	221,935
	Unseen	15,758	79,918	804,404	123,479
	Total	**1,101,504**	**22,169,689**	**3,660,507**	**812,740**

Baseline and Setting

- We compare our method with 9 unsupervised models
  - Single *f*(.) functions: UF, IO, and CP
  - Betweenness Centrality (BC)
  - Jaccard Coefficient (JC)
  - Preferential Attachment (PA)
  - Attractiveness (AT)*
  - PageRank with Priors (PRP)
  - AT-PRP
- Base inference method: Loopy Believe Propagation (LBP)
- Evaluation metrics
  - Area Under ROC Curve (AUC)
  - Normalized Discounted Cumulative Gain (NDCG)

* H.-H. Wu and M.-Y. Yeh, **Influential Nodes in One-Wave Diffusion Model for Location-Based Social Networks**, *PAKDD-2013*

Result

Method	Foursquare		Twitter		Plurk		DBLP	
	AUC	NDCG	AUC	NDCG	AUC	NDCG	AUC	NDCG
UF	76.74	21.66	73.49	18.87	71.08	35.01	70.28	25.07
IO	81.31	51.60	69.98	18.93	69.86	35.33	68.51	23.84
CP	74.03	20.56	67.38	17.15	70.69	36.13	69.52	24.22
BC	67.01	21.26	67.65	18.97	69.81	31.47	64.17	21.10
JC	64.30	26.75	65.65	21.05	70.05	35.40	69.96	28.24
PA	72.28	27.09	62.30	16.39	67.42	32.68	71.41	26.12
AT	82.57	44.54	76.95	20.28	69.62	39.29	70.95	28.48
PRP	57.27	17.93	62.41	16.56	69.12	33.64	61.83	21.25
AT-PRP	71.06	22.38	68.17	18.11	70.99	36.03	67.86	24.27
INFER	86.77	70.60	79.11	24.80	74.23	40.24	86.84	41.75
LEARN	**98.61**	**80.44**	**81.29**	**25.87**	**74.42**	**42.61**	**87.29**	**41.84**
Improve	**16.04**	**28.84**	**4.34**	**4.82**	**3.34**	**3.32**	**15.88**	**13.36**

Learning + inference > inference > baselines

Conclusion

- Dealing with data without labels is critical in Big Data era
  - high velocity implies sparse labels or even no label, and human labeling is expensive.
  - Labels might not be available due to privacy concern
- We might be able to do something by incorporating new learning models into existing frameworks