

# Hyperlink Classification via Structured Graph Embedding

G. Lee, S. Kang, and J. J. Whang\*

*ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*  
July 2019 (\*corresponding author)

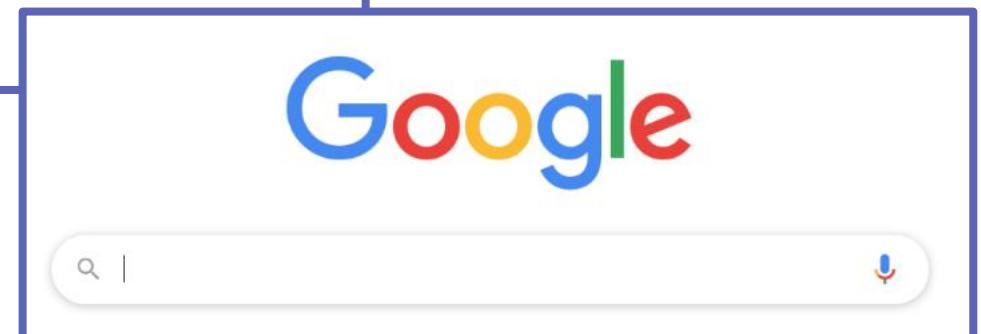
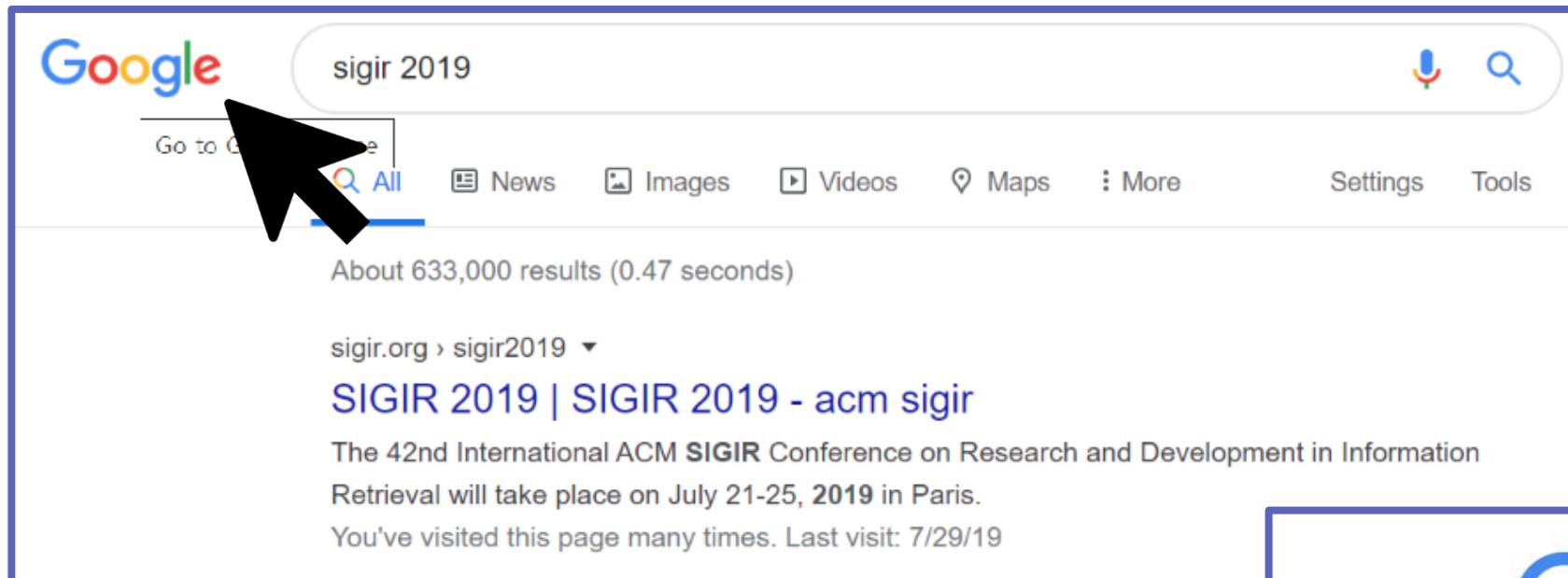
# Real-World Web Graphs

---

- Hyperlinks are created for different reasons
  - **Navigation links**: navigate the main website
  - **Suggestion links**: suggest users to take a look at related information
  - **Action links**: invoke actions such as ‘edit’, ‘share’, or ‘send an email’
- **Hyperlink Classification Problem**
  - Classify hyperlinks into three classes: navigation, suggestion, and action

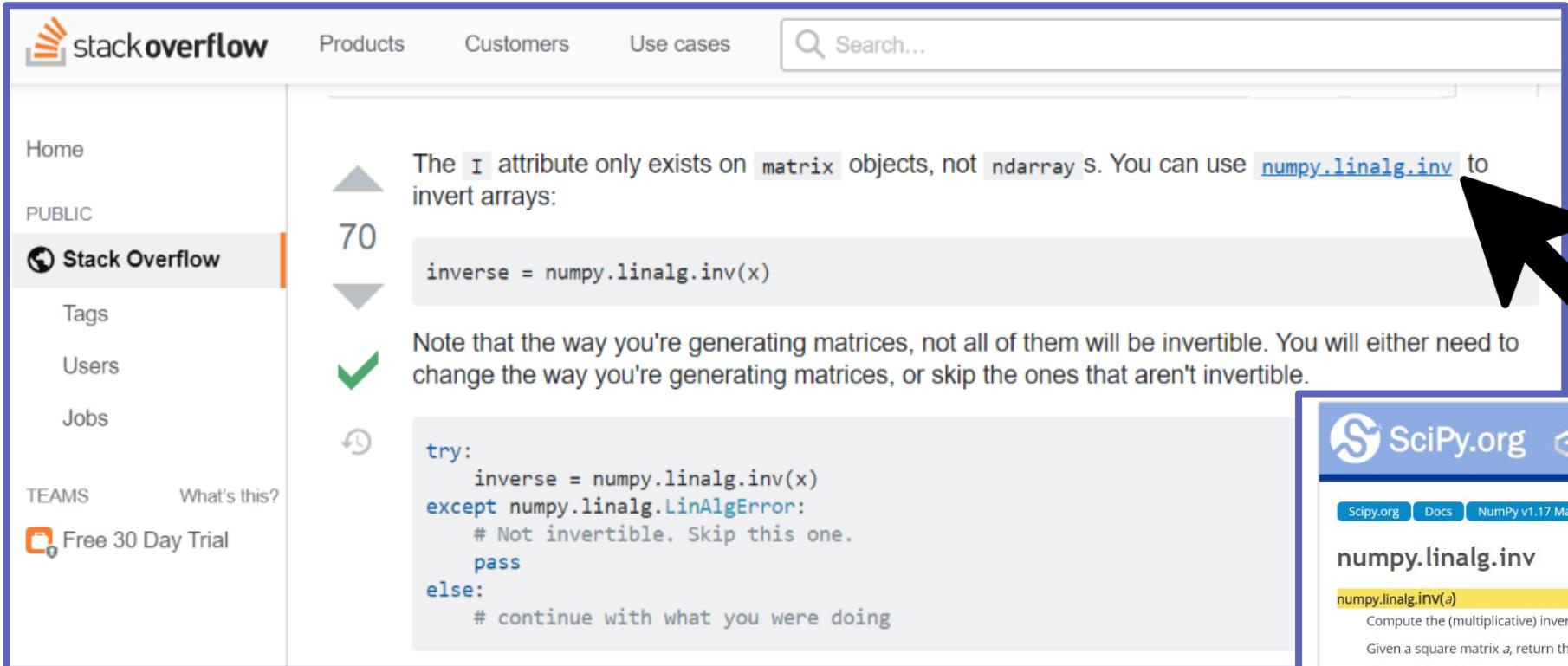
# Hyperlink Classification

## ▪ Navigation Links



# Hyperlink Classification

## Suggestion Links



The screenshot shows a Stack Overflow post. The title of the question is: "The `I` attribute only exists on `matrix` objects, not `ndarray`s. You can use [numpy.linalg.inv](#) to invert arrays:". The post has 70 upvotes. Below the question, there is a code snippet:

```
inverse = numpy.linalg.inv(x)
```

Followed by a note: "Note that the way you're generating matrices, not all of them will be invertible. You will either need to change the way you're generating matrices, or skip the ones that aren't invertible."

Below the note, there is a code block:

```
try:  
    inverse = numpy.linalg.inv(x)  
except numpy.linalg.LinAlgError:  
    # Not invertible. Skip this one.  
    pass  
else:  
    # continue with what you were doing
```



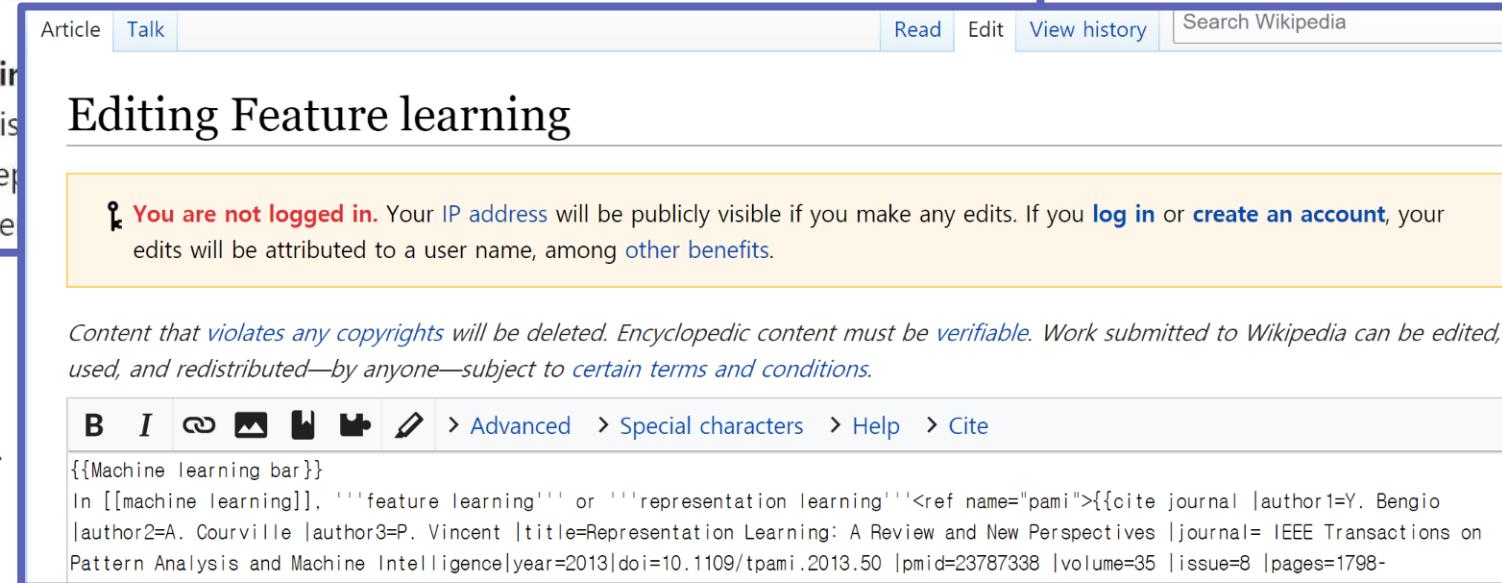
The screenshot shows the SciPy.org documentation for the `numpy.linalg.inv` function. The page title is "numpy.linalg.inv". The documentation states: "Compute the (multiplicative) inverse of a matrix." It describes the function as returning the matrix `ainv` satisfying  $\text{dot}(a, \text{ainv}) = \text{dot}(\text{ainv}, a) = \text{eye}(a.\text{shape}[0])$ . The parameters are `a`:  $(..., M, M)$  array\_like, which is a matrix to be inverted. The returns are `ainv`:  $(..., M, M)$  ndarray or matrix, which is the (Multiplicative) inverse of the matrix `a`. The raises section lists `LinAlgError` if `a` is not square or inversion fails.

# Hyperlink Classification

## ■ Action Links



The screenshot shows a Wikipedia article page for "Feature learning". The page title is "Feature learning" and it is from the English Wikipedia. The page content discusses feature learning in machine learning, stating that it allows a system to automatically discover useful features from raw data. A large black arrow points from the bottom left towards the "Edit" button in the top right corner of the main content area.



The screenshot shows the "Edit" section of the same Wikipedia article. It includes a warning message: "You are not logged in. Your IP address will be publicly visible if you make any edits. If you log in or create an account, your edits will be attributed to a user name, among other benefits." Below this, there is a note about content policies: "Content that violates any copyrights will be deleted. Encyclopedic content must be verifiable. Work submitted to Wikipedia can be edited, used, and redistributed—by anyone—subject to certain terms and conditions." At the bottom, there is a toolbar with various editing icons and links to "Advanced", "Special characters", "Help", and "Cite".

# Real-World Datasets

---

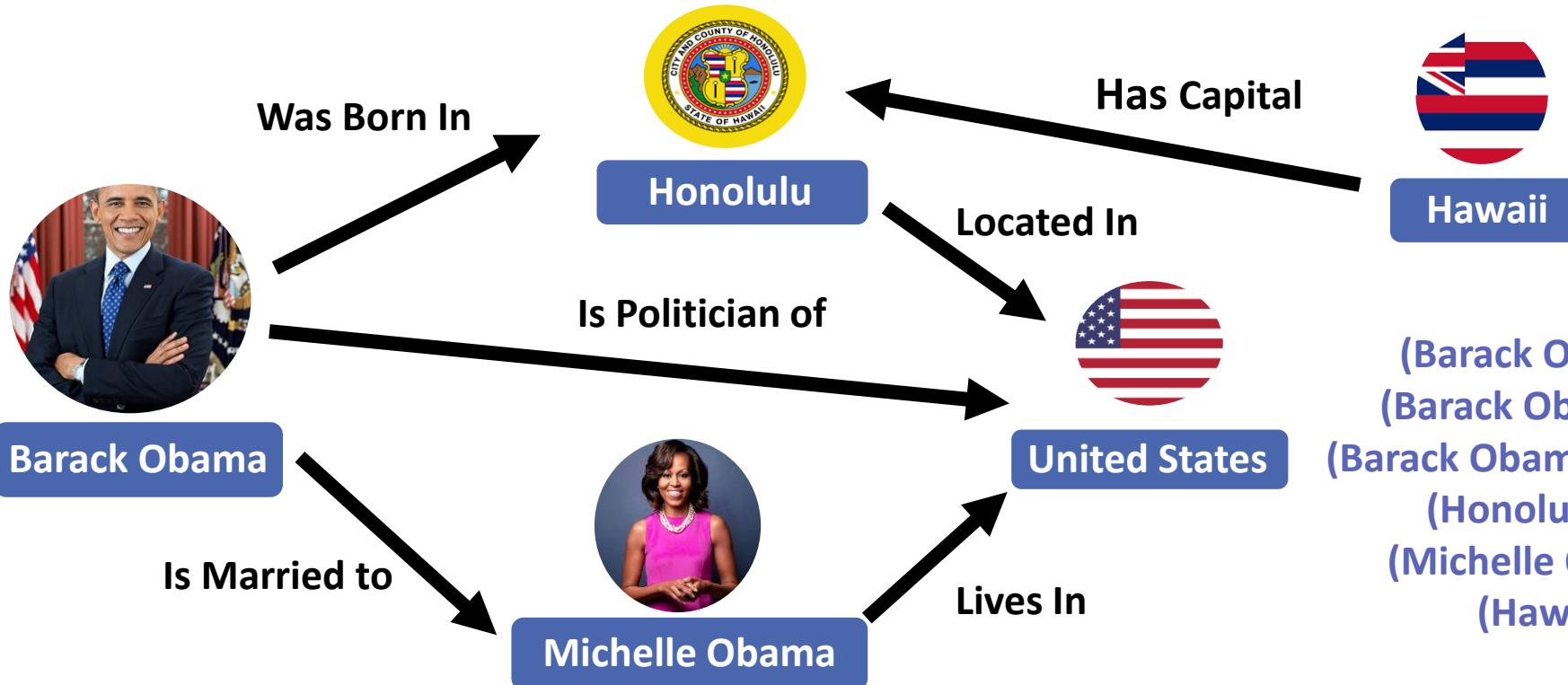
- Real-world web graphs
  - **Crawling** a set of web pages and hyperlinks starting from a page in Stack Overflow.
  - Conducting a **biased random walk**

	$ \mathcal{V} $	$ \mathcal{E} $	<i>navigation</i>	<i>suggestion</i>	<i>action</i>
web_437	404	437	268 (61.33%)	112 (25.63%)	57 (13.04%)
web_1442	332	1,442	1,284 (89.04%)	93 (6.45%)	65 (4.51%)
web_10000	2,202	10,000	9,892 (98.92%)	85 (0.85%)	23 (0.23 %)

**web\_437 and web\_1442:** some heuristics are applied to balance the class sizes.  
**web\_10000** reflects the underlying distribution of the class sizes – very unbalanced.

# Knowledge Graphs

- Graphical Representation of Human Knowledge
  - Each fact is represented by a triplet (**head entity, relation, tail entity**)



(Barack Obama, **was born in**, Honolulu)  
(Barack Obama, **is politician of**, Honolulu)  
(Barack Obama, **is married to**, Michelle Obama)  
(Honolulu, **located in**, United States)  
(Michelle Obama, **lives in**, United States)  
(Hawaii, **has capital**, Honolulu)

# Knowledge Graph Embedding

---

- **Representation Learning Technique**
  - Represents entities and relations in **a feature space**.
  - Given a set of **golden triplets** ( $S$ ) and a set of **corrupted triplets** ( $S'$ ), minimize

$$L = \sum_{(h,r,t) \in S} \sum_{(h',r,t') \in S'} [f(h, r, t) + \gamma - f(h', r, t')]_+$$

How to compute  $f(h, r, t)$  determines different embedding models.

# Knowledge Graph Embedding

- **Knowledge Graph Embedding Models**
  - **TransE:** Translating Embeddings for Modeling Multi-relational Data
  - **TransH:** Knowledge Graph Embedding by Translating on Hyperplanes
  - **TransR:** Learning Entity and Relation Embeddings for Knowledge Graph Completion

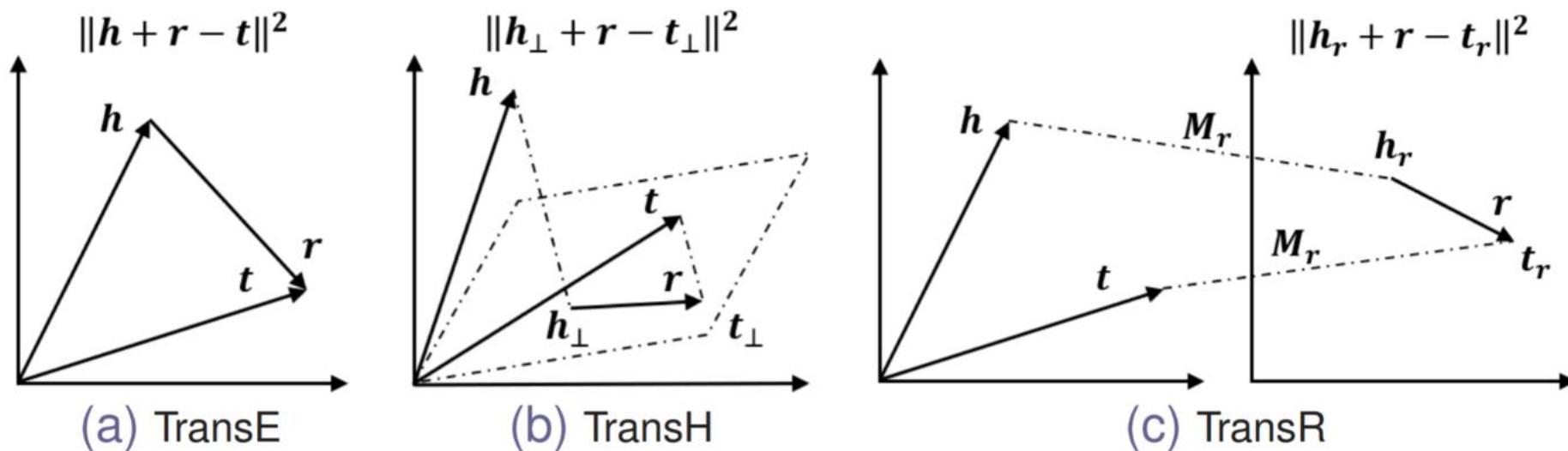
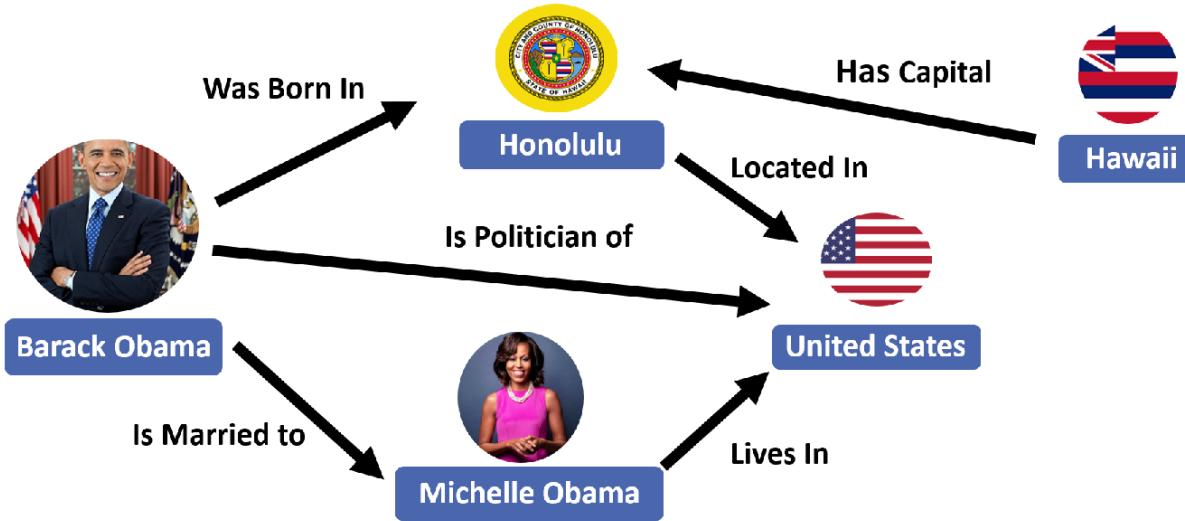


Image from “Knowledge graph embedding: A survey of approaches and applications.” TKDE 2017.

# Hyperlink Classification Model

- Interpret a **Web Graph** as a **Knowledge Graph**

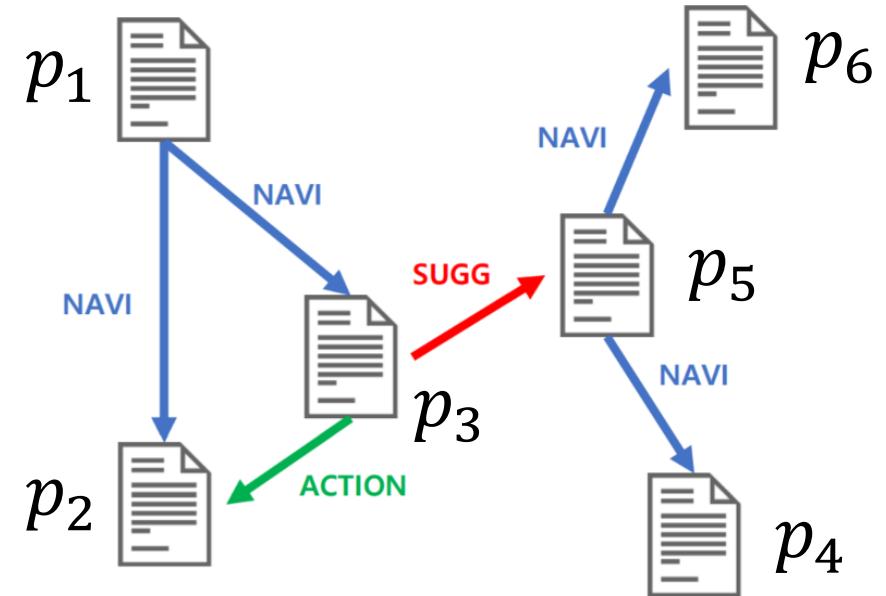


(Barack Obama, was born in, Honolulu)

(Honolulu, located in, United States)

(Michelle Obama, lives in, United States)

⋮



( $p_1$ , NAVI,  $p_3$ )

( $p_3$ , ACTION,  $p_2$ )

( $p_3$ , SUGG,  $p_5$ )

⋮

# Hyperlink Classification Model

---

- Model Specification and Training
  - A web graph  $G = (\mathcal{V}, \mathcal{E})$  where  $\mathcal{V} = \{p_1, p_2, \dots, p_n\}$ ,  $\mathcal{E} = \{(p_i, p_j) : p_i \in \mathcal{V}, p_j \in \mathcal{V}\}$
  - Each hyperlink has one of the three relation labels  $\mathcal{R} = \{n, s, a\}$

$$L = \sum_{(p_i, r, p_j) \in \mathcal{S}} [f(p_i, r, p_j) + \gamma - f(c(p_i, r, p_j))]_+$$

where  $c(p_i, r, p_j)$  is defined by

$$c(p_i, r, p_j) = \begin{cases} \text{prob. } \alpha/2 : & (p_i, r, q), q \in \mathcal{V} \setminus \{p_j\}, (p_i, r, q) \notin \mathcal{S} \\ \text{prob. } \alpha/2 : & (q, r, p_j), q \in \mathcal{V} \setminus \{p_i\}, (q, r, p_j) \notin \mathcal{S} \\ \text{prob. } (1 - \alpha) : & (p_i, r', p_j), r' \in \mathcal{R} \setminus \{r\} \end{cases}$$

$\alpha$  controls the chance to corrupt entities ( $0 < \alpha \leq 1$ )

# Hyperlink Classification Model

---

## ■ Prediction

- For a directed edge  $(p_i, p_j)$  in a test set, the **relation label** is predicted by

$$r^* = \underset{r \in R}{\operatorname{argmin}} f(p_i, r, p_j)$$

- For TransH embedding model,  $f(p_i, r, p_j)$  is computed by

$$f(p_i, r, p_j) = \|(p_i - w_r^T p_i w_r) + r - (p_j - w_r^T p_j w_r)\|_2^2$$

$p_i$  and  $p_j$  : embedding vectors of the pages

$r$  : embedding vector for the relation

$w_r$  : norm vector of the relation-specific hyperplane

# Experimental Results

- F1 scores (%) of our model with different  $\alpha$  values and the original TransE, TransH, and TransR.

		TransE	TransH	TransR
web_437	Our model, $\alpha = 0.3$	34.29	<b>60.25</b>	57.99
	Our model, $\alpha = 0.5$	34.39	58.87	57.32
	Our model, $\alpha = 0.7$	33.88	58.91	<b>59.83</b>
	The original model	<b>36.22</b>	54.04	53.22
web_1442	Our model, $\alpha = 0.3$	23.39	53.42	<b>50.04</b>
	Our model, $\alpha = 0.5$	<b>24.86</b>	<b>55.16</b>	46.18
	Our model, $\alpha = 0.7$	21.18	52.70	45.12
	The original model	20.05	29.94	10.35
web_10000	Our model, $\alpha = 0.3$	<b>20.68</b>	<b>76.00</b>	<b>53.86</b>
	Our model, $\alpha = 0.5$	17.98	74.64	46.99
	Our model, $\alpha = 0.7$	19.50	72.94	44.11
	The original model	15.31	25.35	2.08

→ Our model significantly outperforms the original knowledge graph embedding methods.

→ Creating corrupted triplets by relation perturbation plays a critical role in the hyperlink classification problem.

# Experimental Results

- F1 score (%) of each class and the average F1 score

		<i>navigation</i>	<i>suggestion</i>	<i>action</i>	Average
web_437	Random-predict	59.75	25.81	11.07	32.21
	Rule-based	60.20	20.96	0.00	27.05
	TransE-original	55.78	31.96	20.93	36.22
	TransH-original	70.80	52.75	38.56	54.04
	TransR-original	67.87	52.86	38.94	53.22
web_1442	Our Model	<b>77.04</b>	<b>57.05</b>	<b>46.64</b>	<b>60.25</b>
	Random-predict	89.13	5.18	5.65	33.32
	Rule-based	72.98	10.20	36.67	39.95
	TransE-original	42.54	8.57	9.05	20.05
	TransH-original	54.80	13.57	21.45	29.94
web_10000	TransR-original	0.00	12.97	18.09	10.35
	Our Model	<b>93.48</b>	<b>22.88</b>	<b>49.12</b>	<b>55.16</b>
	Random-predict	98.91	1.60	0.00	33.50
	Rule-based	68.81	1.74	9.92	26.82
	TransE-original	43.25	2.06	0.61	15.31
	TransH-original	63.01	12.02	1.03	25.35
	TransR-original	0.00	5.61	0.61	2.08
	Our Model	<b>99.66</b>	<b>83.22</b>	<b>45.12</b>	<b>76.00</b>

→ Random-predict: random prediction while preserving the number of hyperlinks in each class.

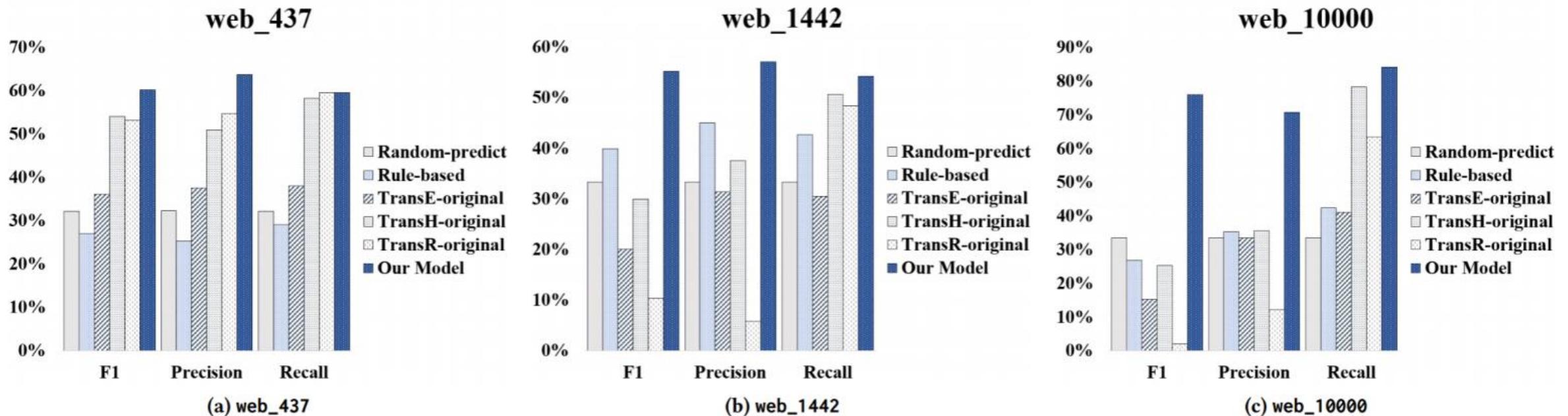
→ Rule-based:

- navigation: within-domain hyperlinks
- action: ‘edit’, ‘share’, ‘email’, or ‘vote’
- suggestion: the rest

→ Our model achieves the highest F1 scores.

# Experimental Results

- The average F1, average precision, and average recall



# Experimental Results

---

- Performance on the original web graphs and the randomly shuffled graphs

		<i>navigation</i>	<i>suggestion</i>	<i>action</i>
<b>web_437</b>	Original Graph	77.04	57.05	46.64
	Randomly Shuffled Graph	58.60	25.36	13.79
<b>web_1442</b>	Original Graph	93.48	22.88	49.12
	Randomly Shuffled Graph	86.08	6.19	5.68
<b>web_10000</b>	Original Graph	99.66	83.22	45.12
	Randomly Shuffled Graph	98.43	1.28	0.61

Randomly shuffled graph: the relation labels are randomly shuffled.

Classification performance significantly degrades on the randomly shuffled graphs.

Real-world web graphs have characterized structures in terms of forming each relation type.

→ Enables us to predict the relation labels via structured graph embedding.

# Summary

---

- **Hyperlink Classification** in Web Search
  - Classify hyperlinks into three classes: **navigation, suggestion, and action**
- Approach the problem from a **structured graph embedding** perspective
  - Interpret a **web graph** as a **knowledge graph**
  - Modify knowledge graph embedding techniques
- **Relation perturbation in negative sampling** enables us to significantly improve performance in classifying hyperlinks on web graphs.

More Information: <http://bigdata.cs.skku.edu/>