# ColdRoute: Effective Routing of Cold Questions in Stack Exchange Sites

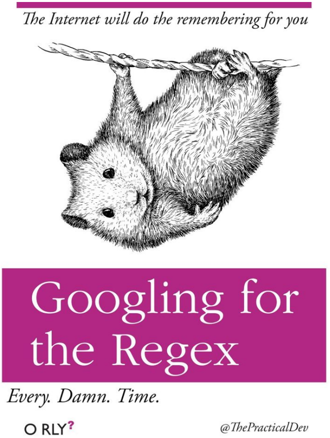
Jiankai Sun [1]
Abhinav Vishnu [2]   Aniket Chakrabarti [3]   Charles Siegel [2]
Srinivasan Parthasarathy [1]

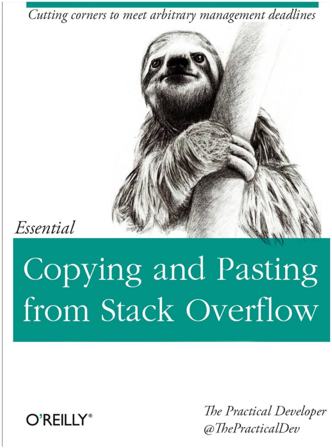
[1]The Ohio State University

[2]Pacific Northwest National Laboratory [3]Microsoft

ECML PKDD, Sep 2018

# Object-Oriented-Programming vs StackOverflow-Oriented-Programming



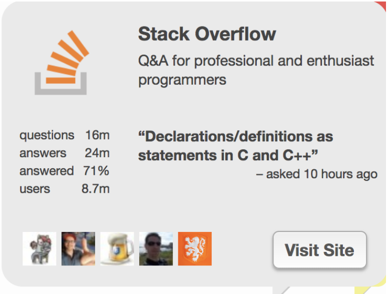




0 Left pic: `https://goo.gl/5vKuaR`

0 Right pic: `https://goo.gl/XAG4DP`

# Motivation

- What if you could not find answers for your questions in Stack Overflow? 4.8m unanswered
- Cold questions: newly posted questions without answer (cold questions) asked by new registered (cold askers) or existing askers
- Cold-Start Problem: Find the right experts to answer cold questions



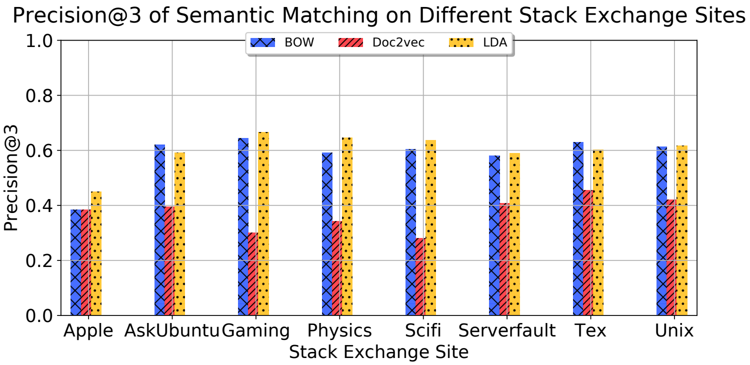

[0] Pic: `https://goo.gl/3VEsV9`

# Related Work: Semantic Matching Models

- Leverage textual information to route cold questions
  - question $\rightarrow$ (answer with the highest semantic similarity) $\rightarrow$ best answerer
  - cold question $\rightarrow$ user with the highest semantic similarity
- How to model textual information?
  - BOW : bag of words [Zhou et al. 2012; 2013; Figueroa and Neumann 2013;]
  - LDA: Latent Dirichlet allocation, Topic Modeling [Guo et al. 2008; Ji et al. 2012]
  - Doc2Vec: Distributed Representations of Sentences and Documents [Le and Mikolov 2014; Dong et al. 2015]

# Finding best answers by semantic matching

- $Precision@3$ computes the average number of times that the best answer is ranked top-$3$ by a specific method



# Challenge I:
## Find the indicator of the best answerer

Up Vote

3

Down Vote

1. MLP is sensitive to feature scaling. Have you normalzied your data?
2. Modify your network structure: add more hidden layers and change number of perceptrons of each layer
3. change activation function to sigmod/tanh/relu etc.

share improve this answer

answered Jun 5 '17 at 18:07

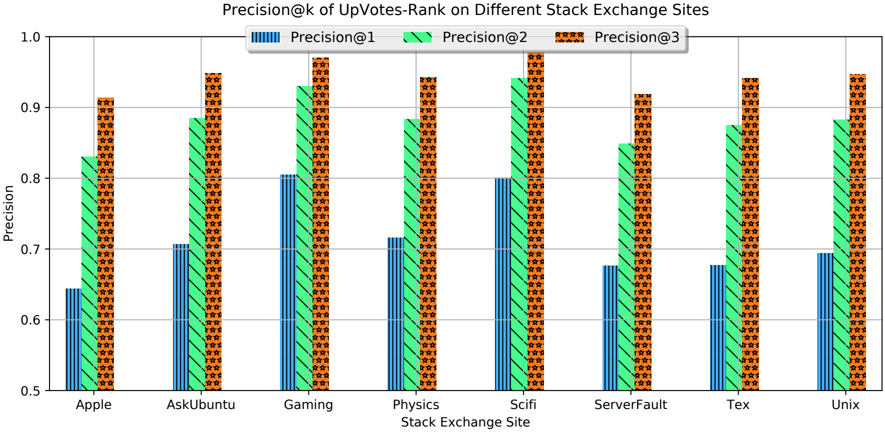
zhenv5

103 ● 4

4. Change learning rate: `learning_rate`, `learning_rate_init`. 5. Toggle `early_stopping` – ijoseph Apr 10 at 22:13

How about voting score (up-votes - down-votes)?

# UpVotes-Rank: select the answerer with the highest voting score as the best answerer



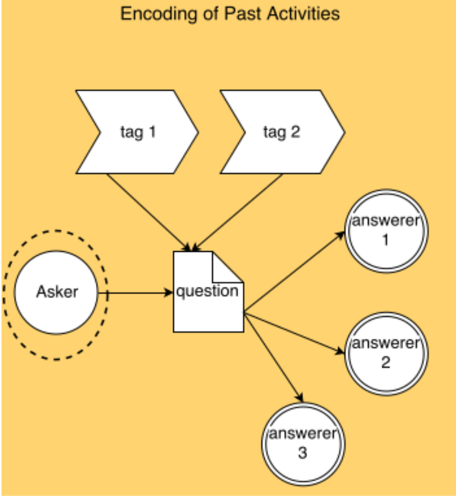
# Model it as A Regression Problem

- How to identify the best answerer for a newly posted question?
  - Model the cold-start problem as a regression problem?

- Predict voting score for each question-answerer pair
- Select the user who has the highest voting score as the best answerer

# Challenge II:

## How to encode past activities as feature vectors?



# Encode Past Activities by One-hot Encoding

| | Feature Vector $\vec{X}$ | | | | | | | | | | | | | | | | Target $\vec{y}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\vec{x}^{(1)}$ | 0 | 0 | 1 | $\cdots$ | 1 | 0 | 0 | $\cdots$ | 1 | 0 | 0 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 4 | $y^{(1)}$ |
| $\vec{x}^{(2)}$ | 0 | 0 | 1 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 1 | 0 | 0 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 3 | $y^{(2)}$ |
| $\vec{x}^{(3)}$ | 0 | 0 | 1 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 1 | 0 | 0 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 2 | $y^{(3)}$ |
| $\vec{x}^{(4)}$ | 0 | 1 | 0 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 5 | $y^{(4)}$ |
| $\vec{x}^{(5)}$ | 0 | 1 | 0 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 6 | $y^{(5)}$ |
| $\vec{x}^{(6)}$ | 1 | 0 | 0 | $\cdots$ | 1 | 0 | 0 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 1 | 1 | 1 | $\cdots$ | 2 | $y^{(6)}$ |
| $\vec{x}^{(7)}$ | 1 | 0 | 0 | $\cdots$ | 0 | 0 | 1 | $\cdots$ | 0 | 1 | 0 | $\cdots$ | 1 | 1 | 1 | $\cdots$ | 4 | $y^{(7)}$ |
| | $q_1$ | $q_2$ | $q_3$ | $\cdots$ | $u_1$ | $u_2$ | $u_3$ | $\cdots$ | $a_1$ | $a_2$ | $a_3$ | $\cdots$ | $t_1$ | $t_2$ | $t_3$ | $\cdots$ | | |
| | Question | | | | Answerer | | | | Asker | | | | Question Tags | | | | | |

# Challenge III:

## Feature vector is very sparse

- Each feature vector $\vec{x}^{(i)}$ has only $(3 + ||\vec{t_i}||_1)$ ones.
- $||\vec{t_i}||_1$ represents question $q_i$'s number of tags (number of ones in the vector $\vec{t_i}$).
- Average number of tags per question in our experiments is $2.5$

# Advantages of Factorization Machine [Rendle et al. 2010, 2012]

- Regression model: $\hat{\vec{y}}(\vec{x}) = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} \sum_{j=i+1}^{p} x_i x_j < \vec{v}_i, \vec{v}_j >$
- Can handle sparse settings very well in comparison with other regressors such as linear regression
- Gives us the flexibility to explore the different features' relative importance in cold question routing (feature selection)
  - asker
  - tags
  - textual descriptions: question head and question body

# A Toy Example

- Given a cold question $q_4$ asked by a new asker $a_4$ with tags $t = \{t_1, t_2, t_3\}$, Predict voting score of $u_3$?

- Linear Regression

$$\hat{y}(\vec{x}) = \boxed{w_0 + w_{q_4} + w_{u_3} + w_{a_4} + \sum_{i=1}^{3} w_{t_i}} + \sum_{i \in \boldsymbol{S}} \sum_{j \in \boldsymbol{S}, i<j} \langle \vec{v}_i, \vec{v}_j \rangle$$

- where $w_0 \in \mathbb{R}, \vec{w} \in \mathbb{R}^p, \vec{V} \in \mathbb{R}^{p \times k}$

# A Toy Example

- Given a cold question $q_4$ asked by a new asker $a_4$ with tags $t = \{t_1, t_2, t_3\}$, Predict voting score of $u_3$?

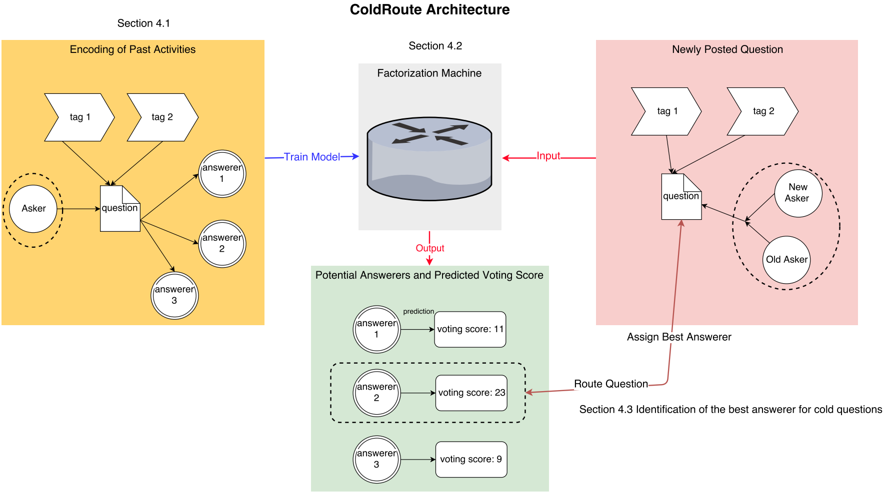
- Linear Regression

$$\hat{y}(\vec{x}) = \boxed{w_0 + w_{q_4} + w_{u_3} + w_{a_4} + \sum_{i=1}^{3} w_{t_i}} + \sum_{i \in \boldsymbol{S}} \sum_{j \in \boldsymbol{S}, i<j} \langle \vec{v}_i, \vec{v}_j \rangle$$

- Interactions among question, asker, answerer, and tags $\boldsymbol{S} = \{q_4, u_3, a_4, t_1, t_2, t_3\}$

- where $w_0 \in \mathbb{R}, \vec{w} \in \mathbb{R}^p, \vec{V} \in \mathbb{R}^{p \times k}$

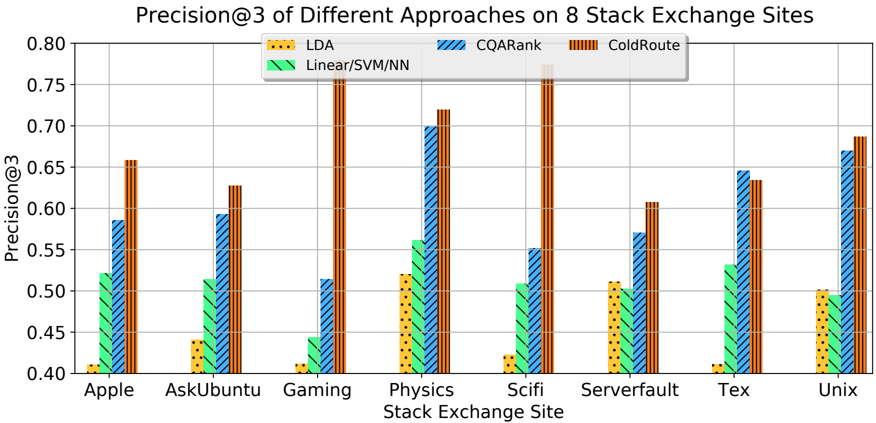
# Putting it all together: Architecture of ColdRoute



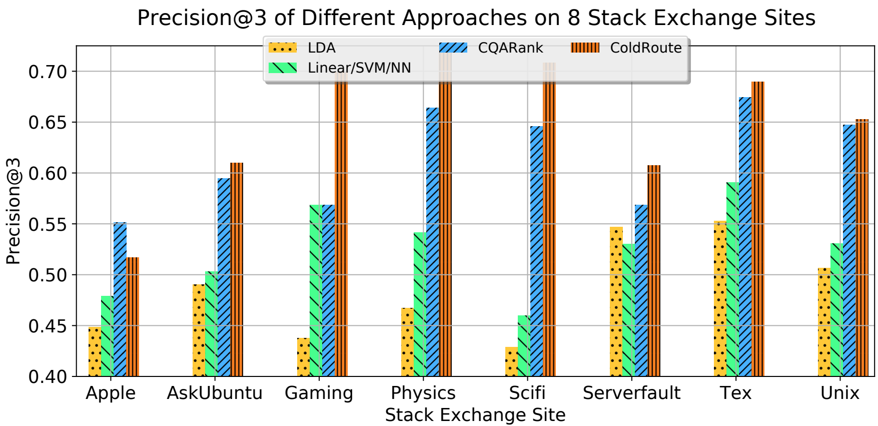
# Experiments

- Datasets: **8** Stack Exchange Sites (Apple, Physics, Gaming, etc)
- Evaluation metrics: Mean Reciprocal Rank (MRR), Precision@k, and Accuracy
- Two different kinds of cold questions:
  - asked by new registered users
  - asked by existing users (have asked questions before)

# Routing Cold Questions Asked by Existing Askers (ColdRoute vs state-of-the-arts)



# Routing Cold Questions Asked by New Askers (ColdRoute vs state-of-the-arts)



# Feature Selection for Routing Cold Questions Asked by New Askers (Evaluated by $Precision@3$)

| | Apple | Ask. | Gaming | Physics | Scifi |
|---|---|---|---|---|---|
| BOW | 0.3840 | 0.4357 | 0.3000 | 0.3799 | 0.2484 |
| Doc2Vec | 0.3840 | 0.4096 | 0.3563 | 0.3493 | 0.2547 |
| LDA | 0.4487 | 0.4902 | 0.4375 | 0.4672 | 0.4286 |
| ColdRoute-HB | 0.5133 | 0.4989 | 0.4625 | 0.5109 | 0.4534 |
| ColdRoute-B | 0.4829 | 0.5139 | 0.4563 | 0.5284 | 0.3975 |
| ColdRoute-H | 0.4829 | 0.5468 | 0.5063 | 0.5633 | 0.4907 |
| ColdRoute-T | 0.5171 | 0.6100 | 0.7000 | 0.7249 | 0.7081 |

# Conclusion and Future Work

## Conclusion

- Propose **ColdRoute** for tacking cold questions routing in CQAs
- Overcome **3** challenges
  - Use voting score as the indicator of the best answerer
  - Encode users' past activities by one-hot encoding
  - Address sparse settings by Factorization Machine (**FM**)
- Simple modes are good like **FM** which are proven to be better than simple DL algorithms
- Code: `https://github.com/zhenv5/ColdRoute`

## Future work

- Improve over the state-of-the-art models a lot, Still have improvement space
- Combine the power of deep neural network and factorization machines

# Acknowledgments

- **Acknowledgments:** This work is supported by NSF grants CCF-1645599, IIS-1550302, and CNS-1513120, and a grant from the Ohio Supercomputer Center (PAS0166). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their sponsors.

# Q & A

THE OHIO STATE UNIVERSITY

# Appendix

# FM as a regressor

Consider a $2$-way FM ($d = 2$) as an example:

$$\hat{\vec{y}}(\vec{x}) = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} \sum_{j=i+1}^{p} x_i x_j < \vec{v}_i, \vec{v}_j > \quad (1)$$

where the model parameters that have to be estimated are:

$$w_0 \in \mathbb{R}, \vec{w} \in \mathbb{R}^p, \vec{V} \in \mathbb{R}^{p \times k} \quad (2)$$

And $< \cdot, \cdot >$ is the dot product of two vectors of size $k$:

$$< \vec{v}_i, \vec{v}_j >= \sum_{f=1}^{k} v_{i,f} v_{j,f} \quad (3)$$

where a row $\vec{v}_i \in \vec{V}$ describes the $i$-th variable with $k \in \mathbb{N}_0^+$ factors. $k$ represents the dimensionality of the factorization.

# Gradient descent to update parameters

$$\frac{\partial}{\partial\theta}\hat{\vec{y}}(\vec{x}) = \begin{cases} 1, & \text{if } \theta \text{ is } w_0 \\ x_i, & \text{if } \theta \text{ is } w_i \\ x_i \sum_{j=1}^{p} v_{j,f} x_j - v_{i,f} x_i^2, & \text{if } \theta \text{ is } v_{i,f} \end{cases} \tag{4}$$

# Evaluation Metric

## MRR.

The MRR measure is given by

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{r^q_{best}} \tag{5}$$

where $r^q_{best}$ is the position of question $q$'s best answerer in the predicted ranking list. It's worth mentioning that MRR is equivalent to Mean Average Precision (MAP) since the number of correct elements (the best answerer) in the predicted ranking list is just $1$.

# Evaluation Metric

## Precision@$k$.

The $Precision@k$ is applied to measure the average number of times that the best answerer is ranked on top-$k$ by a certain algorithm.

$$Precision@k = \frac{\{q \in Q | r_{best}^{q} <= k\}}{|Q|} \tag{6}$$

**Accuracy.** The Accuracy is used to measure the ranking quality of the best answerer, given by

$$Accuracy = \frac{1}{|Q|} \sum_{q \in Q} \frac{|R^q| - r_{best}^{q}}{|R^q| - 1} \tag{7}$$

Where $Accuracy = 1$ (best) means that the best answerer returned by an algorithm always ranks on top while $Accuracy = 0$ means the opposite.

# Statistics of Stack Exchange Sites (Ask., Ser. are short for AskUbuntu and Serverfault respectively)

| | Apple | Ask. | Gaming | Physics | Scifi | Ser. | Tex | Unix |
|---|---|---|---|---|---|---|---|---|
| # Questions | 80,466 | 257,173 | 75,696 | 93,529 | 38,026 | 238,764 | 129,182 | 111,505 |
| # Answers | 119,878 | 337,198 | 130,294 | 137,258 | 78,652 | 398,470 | 169,354 | 171,016 |
| # Unique Users | 65,851 | 189,955 | 51,192 | 41,115 | 26,673 | 130,951 | 48,049 | 65,279 |
| # Questions having Best Answers | 29,765 | 85,843 | 45,798 | 38,094 | 21,740 | 117,275 | 76,862 | 53,856 |
| # Unique Tags | 1,048 | 3,020 | 4,437 | 876 | 2,349 | 3,514 | 1,525 | 2,438 |
| Avg # Tags per Question | 2.824 | 2.6982 | 1.2823 | 2.9634 | 2.1967 | 2.882 | 2.2752 | 2.7868 |
| # Askers | 40,206 | 137,171 | 25,153 | 31,415 | 12,413 | 93,739 | 42,819 | 45,773 |
| # Asker (asked only 1 question) (%) | 76.74% | 75.88% | 74.23% | 63.26% | 74.71% | 64.04% | 62.55% | 68.48% |
| Avg # Questions per Asker | 1.9758 | 1.8557 | 2.9689 | 2.8849 | 3.0031 | 2.4411 | 2.9851 | 2.4022 |