Learning Search Tasks in Queries and Web Pages via Graph Regularization

Ming Ji, Jun Yan, Siyu Gu, Jiawei Han, Xiaofei He, Wei Vivian Zhang, Zheng Chen
University of Illinois at Urbana-Champaign
mingji1@illinois.edu
Problem: Search task prediction

- Ultimate goal of search engines: help users accomplish their search tasks
  - Book flight
  - Exchange currency
  - Check the weather
  - And more…
Problem: Search task prediction

- Currently, the search engines can handle some popular search tasks—flight Beijing Chicago, weather Beijing ...

- But those task-oriented functions only work for queries with specific keywords—Beijing rain

- We need to find out all the queries having this task!

- What about tasks that cannot be covered by those functions?
  - Thinkpad T410 broken
    - General information about Thinkpad T410?
    - Lenovo troubleshooting page?
  - We also need to find out the tasks behind web pages!
Problem definition

- **Query:** Thinkpad T410 broken
- **Search task:** the action that the user wants to perform towards the entity
  - We mainly work with named entity queries and related web pages
  - The search task is inferred from “other terms”
- **Target:** queries with a certain category of named entities and the related web pages
  - Named entity category: a set of entities that are usually considered to be of the same kind, such as computers, cars, cities, etc.
Problem definition

- **Input data for training**
  - A set of popular search tasks $T = \{t_1, \ldots, t_{|T|}\}$ that we want to study
  - A set of queries $Q = \{q_1, \ldots, q_{|Q|}\}$ with entities of the same category
  - A set of web pages $P = \{p_1, \ldots, p_{|P|}\}$ clicked by different users after issuing these queries
  - A subset of queries $\{q_1, \ldots, q_n\}$ and a subset of web pages $\{p_1, \ldots, p_m\}$ are labeled by the search tasks, $n \leq |Q|$, $m \leq |P|$
Problem definition

- **Output**
  
  - One prediction function $f$ for queries, and one prediction function $g$ for web pages

- **Application**
  
  - Given any unseen query $q$ and any unseen web page $p$ represented by content features, output the following:

    $$
    f(q) = [f^{(1)}(q), ..., f^{(|T|)}(q)]^T, \quad \mathbf{g}(p) = [g^{(1)}(p), ..., g^{(|T|)}(p)]^T
    $$

    $f^{(t)}(q)$: confidence score that $q$ is triggered by task $t$

    $g^{(t)}(p)$: confidence score that $p$ can accomplish task $t$

  - Predict the most probable task behind each query and each web page

    $$
    \text{task}(q) = \arg\max_{1 \leq t \leq |T|} f^{(t)}(q), \quad \text{task}(p) = \arg\max_{1 \leq t \leq |T|} g^{(t)}(p)
    $$
What information do we have

Semi-supervised classification on queries and web pages simultaneously
Challenges

How to organize the information on two types of data (queries and web pages)?
- Different feature spaces
- Content information + click-through information

Solution
- All the information we have
  => task similarity among queries and web pages
  => a task-oriented heterogeneous graph
Content feature extraction

- **Task phrase**: the substring left after removing the terms corresponding to named entities
  
  - Thinkpad T410 broken

- **A task phrase should represent the same task for entities of the same category**

- **Merge queries with the same task phrases to clusters because they share the same tasks**
  
  - “thinkpad T410 broken”, “MacBook Pro broken”, “HP Pavilion dv6z broken” => “* broken”
  
  - The words of the task phrases are used as the task-oriented content features

- **Extract task-oriented terms from web pages similarly**
Learning from multi-typed data

- Content-based task similarity
  - Task phrases/task-oriented terms containing similar words are likely to share similar tasks
  - Measure the task similarity by any distance measurement in the task-oriented feature space
  - Build two $k$-nearest neighbor subgraphs to encode the content-based similarity information

\[
W_{q,ij} = \begin{cases} 
\text{sim}(q_i, q_j), & \text{if } q_i \in N_k(q_j) \text{ or } q_j \in N_k(q_i) \\
0, & \text{otherwise.}
\end{cases}
\]

\[
W_{p,ij} = \begin{cases} 
\text{sim}(p_i, p_j), & \text{if } p_i \in N_k(p_j) \text{ or } p_j \in N_k(p_i) \\
0, & \text{otherwise.}
\end{cases}
\]
Learning from multi-typed data

- **Click-based task similarity**
  - A web page $p$ clicked by the user after issuing query $q$ might be useful in accomplishing the search task behind $q$.
  
  - Build a bipartite click subgraph according to the click-through logs to encode the click-based similarity.

\[
R_{qp,ij} = \begin{cases} 
\# \text{ of clicks, if } p_j \text{ is clicked after issuing } q_i \\
0, & \text{otherwise}
\end{cases}
\]
Learning from multi-typed data

- Unify the content and click-through information on two types of data in a task-oriented graph

Click graph $R_{qp}$

Encode the click-based task similarity

Query graph $W_q$

Web page graph $W_p$

Encode the content-based task similarity
Objective function

- A simple linear regression model
  \[ f^{(t)}(q) = w_q^{(t)} q \quad \text{and} \quad g^{(t)}(p) = w_p^{(t)} p \]

- Task indicator vector for a labeled query task
  \[ \text{phrase } q_i: \ u_i = \begin{bmatrix} u_i^{(1)} & \cdots & u_i^{(|T|)} \end{bmatrix}^T \]
  \[ u_i^{(t)} = \begin{cases} 1, & \text{if } q_i \text{ is labeled to have task } t \\ 0, & \text{otherwise} \end{cases} \]

- Task indicator vector for a labeled web page \( p_j \):
  \[ v_j = \begin{bmatrix} v_j^{(1)} & \cdots & v_j^{(|T|)} \end{bmatrix}^T \]
  \[ v_j^{(t)} = \begin{cases} 1, & \text{if } p_j \text{ is labeled to have task } t \\ 0, & \text{otherwise} \end{cases} \]
Objective function

- **Intuition: consistency over the task-oriented graph**
  - Within each subgraph, the confidence estimations of two objects having each task should be similar if they are linked together, with the edge weight measuring the similarity
  - The confidence estimations of the labeled query task phrases and web pages having each task should be consistent to their labels
Objective function

Minimize the following:

\[ J(\mathbf{w}_q^{(t)}, \mathbf{w}_p^{(t)}) = \sum_{i=1}^{Q} \sum_{j=1}^{P} \sum_{k=1}^{|Q_i|} \sum_{l=1}^{|P_j|} R_{q,p,i,j} \left( \frac{\mathbf{w}_{q}^{(t)T} \mathbf{q}_i}{\sqrt{D_{q,p,i,i}}} - \frac{\mathbf{w}_{p}^{(t)T} \mathbf{p}_j}{\sqrt{D_{p,q,j,j}}} \right)^2 \]

\[ + \lambda_q \sum_{i,j=1}^{Q} W_{q,i,j} \left( \frac{\mathbf{w}_{q}^{(t)T} \mathbf{q}_i}{\sqrt{D_{q,i,i}}} - \frac{\mathbf{w}_{q}^{(t)T} \mathbf{q}_j}{\sqrt{D_{q,i,j}}} \right)^2 \]

\[ + \lambda_p \sum_{i=1}^{P} W_{p,i,j} \left( \frac{\mathbf{w}_{p}^{(t)T} \mathbf{p}_i}{\sqrt{D_{p,i,i}}} - \frac{\mathbf{w}_{p}^{(t)T} \mathbf{p}_j}{\sqrt{D_{p,j,j}}} \right)^2 \]

\[ + \alpha_q \sum_{i=1}^{Q} (\mathbf{w}_{q}^{(t)T} \mathbf{q}_i - u^{(t)}_i)^2 \]

\[ + \alpha_p \sum_{i=1}^{P} (\mathbf{w}_{p}^{(t)T} \mathbf{p}_i - v^{(t)}_i)^2 \]

\[ + \beta_q \left\| \mathbf{w}_q^{(t)} \right\|^2 \]

\[ + \beta_p \left\| \mathbf{w}_p^{(t)} \right\|^2 \]

Parameters to control the importance of different terms

Consistency over the click subgraph

Consistency over the content subgraphs

Consistency with the label information

Tikhonov regularizers to ensure stableness of the solution
Objective function

Normalization

- Construct four diagonal matrices $D_{qp}, D_{pq}, D_q, D_p$

$$D_{qp,ii} = \sum_{j=1}^{\lvert P \rvert} R_{qp,ij}, \quad D_{pq,ii} = \sum_{j=1}^{\lvert Q \rvert} R_{qp,ji}$$

$$D_{q,ii} = \sum_{j=1}^{\lvert Q \rvert} W_{q,ij}, \quad D_{p,ii} = \sum_{j=1}^{\lvert P \rvert} W_{p,ij}$$

- $J \left( w_q^{(t)}, w_p^{(t)} \right) = \lambda_{qp} \sum_{i=1}^{\lvert Q \rvert} \sum_{j=1}^{\lvert P \rvert} R_{qp,ij} \left( \frac{w_q^{(t)T} q_i}{\sqrt{D_{q,ii}}} - \frac{w_p^{(t)T} p_j}{\sqrt{D_{p,jj}}} \right)^2$

$$+ \lambda_q \sum_{i,j=1}^{\lvert Q \rvert} W_{q,ij} \left( \frac{w_q^{(t)T} q_i}{\sqrt{D_{q,ii}}} - \frac{w_q^{(t)T} q_j}{\sqrt{D_{q,jj}}} \right)^2 + \lambda_p \sum_{i=1}^{\lvert P \rvert} W_{p,ij} \left( \frac{w_p^{(t)T} p_i}{\sqrt{D_{p,ii}}} - \frac{w_p^{(t)T} p_j}{\sqrt{D_{p,jj}}} \right)^2$$

- Normalization applied in order to reduce the impact of popularity of objects, preventing the confidence of popular objects having each task from increasing incorrectly.
Solution

- It can be proven that the Hessian matrix of $J \left( w_q^{(t)}, w_p^{(t)} \right)$ is positive definite. So our objective function is strictly convex.

- The unique global minimum can be obtained by solving the following linear system:

\[
\frac{\partial J \left( w_q^{(t)}, w_p^{(t)} \right)}{\partial w_q^{(t)T}} = 0, \quad \frac{\partial J \left( w_q^{(t)}, w_p^{(t)} \right)}{\partial w_p^{(t)T}} = 0
\]
Experiments

Data set

- Computer category
  - 780k distinct queries => 2,268 task phrases, represented by 3,210-d lexical features
  - 36,890 web pages which received ≥ 5 clicks, represented by 8,532-d lexical features

- Car category
  - 7,600k distinct queries => 3,308 task phrases, represented by 2,997-d lexical features
  - 33,039 web pages which received ≥ 20 clicks, represented by 11,926-d lexical features
## Experiments

Popular search tasks to be predicted in the computer category

<table>
<thead>
<tr>
<th>Search task</th>
<th>Example task phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchase computer</td>
<td>* amazon, * coupon, * deal</td>
</tr>
<tr>
<td>Find review</td>
<td>* reviews, * desktop, * models</td>
</tr>
<tr>
<td>Compare</td>
<td>* versus *, * or *</td>
</tr>
<tr>
<td>System help</td>
<td>* tech support, * configuration, * recovery</td>
</tr>
<tr>
<td>Download software</td>
<td>* driver download, * audio driver, * media</td>
</tr>
<tr>
<td>Maintain hardware</td>
<td>* broken, * memory upgrade, upgrade * hard drive</td>
</tr>
<tr>
<td>Purchase accessories</td>
<td>* printer, * bag, * camera</td>
</tr>
</tbody>
</table>
## Experiments

Popular search tasks to be predicted in the car category

<table>
<thead>
<tr>
<th>Search task</th>
<th>Example task phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchase a new car</td>
<td>* price, * dealer, * retail</td>
</tr>
<tr>
<td>Purchase a used car</td>
<td>used *, pre-owned *, second hand *</td>
</tr>
<tr>
<td>Find reviews</td>
<td>* reviews, * new model, * cars</td>
</tr>
<tr>
<td>Compare</td>
<td>Compare * and *, * vs *, difference between * and *</td>
</tr>
<tr>
<td>Rent a car</td>
<td>* rental, rent a *, cheap * rental</td>
</tr>
<tr>
<td>Maintain a car</td>
<td>* repair, * problems, * oil leak</td>
</tr>
<tr>
<td>Purchase accessories</td>
<td>wheels for *, * engine, * parts</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>* insurance, * manuals, * troubleshooting</td>
</tr>
</tbody>
</table>
Experiments

- **Algorithms for comparison**

  - Maximum Entropy (ME): supervised, using the content-based features only.

  - LapRLS-content: semi-supervised, using the content-based features only.

  - LapRLS-click: semi-supervised, using the click graph and the content-based features, but do not consider content-based task similarity
## Experiments

### $F_1$ measure on queries of the computer category

<table>
<thead>
<tr>
<th>% labeled queries</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>0.59</td>
<td>0.62</td>
<td>0.66</td>
<td>0.69</td>
<td>0.70</td>
<td>0.71</td>
<td>0.72</td>
<td>0.73</td>
<td>0.68</td>
</tr>
<tr>
<td>LapRLS-content</td>
<td>0.66</td>
<td>0.69</td>
<td>0.73</td>
<td>0.75</td>
<td>0.75</td>
<td>0.76</td>
<td>0.77</td>
<td>0.78</td>
<td>0.73</td>
</tr>
<tr>
<td>LapRLS-click</td>
<td>0.62</td>
<td>0.67</td>
<td>0.71</td>
<td>0.74</td>
<td>0.76</td>
<td>0.77</td>
<td>0.78</td>
<td>0.78</td>
<td>0.73</td>
</tr>
<tr>
<td>Our method</td>
<td>0.67</td>
<td>0.70</td>
<td>0.75</td>
<td>0.77</td>
<td>0.78</td>
<td>0.79</td>
<td>0.80</td>
<td>0.81</td>
<td>0.76</td>
</tr>
</tbody>
</table>

### $F_1$ measure on web pages of the computer category

<table>
<thead>
<tr>
<th>% labeled pages</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>0.39</td>
<td>0.40</td>
<td>0.41</td>
<td>0.41</td>
<td>0.43</td>
<td>0.43</td>
<td>0.43</td>
<td>0.43</td>
<td>0.42</td>
</tr>
<tr>
<td>LapRLS-content</td>
<td>0.45</td>
<td>0.46</td>
<td>0.48</td>
<td>0.46</td>
<td>0.50</td>
<td>0.50</td>
<td>0.51</td>
<td>0.52</td>
<td>0.49</td>
</tr>
<tr>
<td>LapRLS-click</td>
<td>0.53</td>
<td>0.57</td>
<td>0.60</td>
<td>0.61</td>
<td>0.63</td>
<td>0.64</td>
<td>0.64</td>
<td>0.65</td>
<td>0.61</td>
</tr>
<tr>
<td>Our method</td>
<td>0.58</td>
<td>0.62</td>
<td>0.64</td>
<td>0.65</td>
<td>0.67</td>
<td>0.67</td>
<td>0.67</td>
<td>0.68</td>
<td>0.65</td>
</tr>
</tbody>
</table>
## Experiments

### $F_1$ measure on queries of the car category

<table>
<thead>
<tr>
<th>% labeled queries</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>0.50</td>
<td>0.55</td>
<td>0.60</td>
<td>0.62</td>
<td>0.63</td>
<td>0.64</td>
<td>0.65</td>
<td>0.66</td>
<td>0.61</td>
</tr>
<tr>
<td>LapRLS-content</td>
<td>0.69</td>
<td>0.72</td>
<td>0.76</td>
<td>0.77</td>
<td>0.78</td>
<td>0.79</td>
<td>0.79</td>
<td>0.80</td>
<td>0.76</td>
</tr>
<tr>
<td>LapRLS-click</td>
<td>0.68</td>
<td>0.72</td>
<td>0.77</td>
<td>0.79</td>
<td>0.79</td>
<td>0.81</td>
<td>0.81</td>
<td>0.81</td>
<td>0.77</td>
</tr>
<tr>
<td>Our method</td>
<td>0.70</td>
<td>0.74</td>
<td>0.78</td>
<td>0.80</td>
<td>0.81</td>
<td>0.81</td>
<td>0.82</td>
<td>0.82</td>
<td>0.79</td>
</tr>
</tbody>
</table>

### $F_1$ measure on web pages of the car category

<table>
<thead>
<tr>
<th>% labeled pages</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>0.74</td>
<td>0.77</td>
<td>0.78</td>
<td>0.79</td>
<td>0.80</td>
<td>0.80</td>
<td>0.81</td>
<td>0.81</td>
<td>0.79</td>
</tr>
<tr>
<td>LapRLS-content</td>
<td>0.74</td>
<td>0.76</td>
<td>0.79</td>
<td>0.81</td>
<td>0.81</td>
<td>0.81</td>
<td>0.82</td>
<td>0.82</td>
<td>0.79</td>
</tr>
<tr>
<td>LapRLS-click</td>
<td>0.66</td>
<td>0.71</td>
<td>0.75</td>
<td>0.78</td>
<td>0.79</td>
<td>0.80</td>
<td>0.81</td>
<td>0.83</td>
<td>0.77</td>
</tr>
<tr>
<td>Our method</td>
<td>0.75</td>
<td>0.78</td>
<td>0.80</td>
<td>0.82</td>
<td>0.83</td>
<td>0.83</td>
<td>0.84</td>
<td>0.84</td>
<td>0.81</td>
</tr>
</tbody>
</table>
Experiments

- How to use the search task prediction results to improve the search experience?

- Task-oriented re-ranking
  - **Input:** any unseen query \( q \), weight parameter \( \mu \).
  - **Procedure**
    - Retrieve \( k \) relevant web pages \( \{ p_1, ..., p_k \} \) of \( q \) using the original search engine, ranked according to their relevance scores \( \{ r_1, ..., r_k \} \).
    - Predict the search task \( t_q \) behind \( q \), and the confidence estimation \( g^{(t_q)}(p_j) \) that each page \( p_j \) can accomplish \( t_q \), \( j = 1, ..., k \).
    - Normalize \( g^{(t_q)}(p_j) = \frac{g^{(t_q)}(p_j)}{\max\{g^{(t_q)}(p_1), ..., g^{(t_q)}(p_k)\}} \).
    - Compute \( r'_j = r_j + \mu g^{(t_q)}(p_j) \), for each \( j = 1, ..., k \).
    - Re-rank \( p_1, ..., p_k \) in the descending order of \( r'_j \).
  - **Output:** re-ranked web pages \( \{ p'_1, ..., p'_k \} \).
Conclusions

- Propose the definition of search tasks: the action that the user wants to perform towards the entity
  - A finer scale than existing binary or three-class taxonomy of user goals or intents.

- Propose a novel classification algorithm
  - Unify the content and click-through information of both sides of the query-page click relationship into a heterogeneous graph
  - The performance of the classifiers of queries and web pages keep mutually enhancing each other
  - Can predict the search tasks of future queries and web pages solely base on their content
Future work

- How to automatically discover what are the popular search tasks among queries, instead of manual study?
  - Task-oriented clustering?

- How to better consider search tasks in search ranking?
  - Design task-oriented ranking models?