Knowledge is becoming bigger part of Search
Knowledge is becoming a bigger part of Search

Brisbane - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Brisbane

Brisbane is the capital and most populous city in the Australian state of Queensland and the third most populous city in Australia. Brisbane's metropolitan area has a...

Visit Brisbane | Your guide to things to see and do in Brisbane
www.visitbrisbane.com.au

Want to know where to eat, drink, shop, play and stay in Brisbane? What about all the exciting Brisbane events? visitbrisbane is loaded with great ideas on how to...

Brisbane Roar - Scores and Schedule
www.goal.com/en-us

<table>
<thead>
<tr>
<th>Date</th>
<th>Home</th>
<th>Score</th>
<th>Away</th>
<th>League</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mar 17</td>
<td>Central Coast</td>
<td>1 - 0</td>
<td>Brisbane Roar</td>
<td>AUS1</td>
<td>Full Time</td>
</tr>
<tr>
<td>Mar 24</td>
<td>Brisbane Roar</td>
<td>vs</td>
<td>Melbourne Heart</td>
<td>AUS1</td>
<td>2:00 AM ET</td>
</tr>
<tr>
<td>Mar 28</td>
<td>Brisbane Roar</td>
<td>vs</td>
<td>Sydney FC</td>
<td>AUS1</td>
<td>5:00 AM ET</td>
</tr>
</tbody>
</table>

Brisbane travel guide - Wikitravel
wikitravel.org/en/Brisbane

Get in • Do • Buy • Eat • Sleep
Brisbane is the capital of the state of Queensland. It has a population of about 2 million people, making it the third-largest city in Australia.

Brisbane
Brisbane is the capital and most populous city in the Australian state of Queensland and the third most populous city in Australia. Brisbane's metropolitan area has a population of 2.15 million, and...

en.wikipedia.org

Population: 2.15 million (2011)
Date Founded: 1824
Area: 2,297 sq miles (5,950 km²)
Nearby Airports: Brisbane Airport • Archerfield Airport • Eagle Farm Airport • Enoggera Barracks(HLS)

Attractions

Lang Park • South Bank Parklands • Mount Coot-tha, Quee... • Lone Pine Koala San...
Knowledge is becoming bigger part of Search
Knowledge is becoming bigger part of Search
Knowledge is Becoming Part of Search

“… a new breed of search experiences … the user is saved the burden of culling documents from a results list and laboriously extracting information buried within them.”

- Baeza-Yates & Raghavan on Next Generation Web Search

All major search engines have started incorporating “knowledge” into search results

Search users do respond, albeit slowly, to the capabilities of search engines → leading to more innovations on integrating knowledge and search.
Knowledge Graph
Outline

Knowledge extraction
• Finding entities
• Finding relationships
• Finding facts

Search with knowledge
• Query tagging
• Query suggestion

Recent advancements
• Crowdsourcing knowledge
Knowledge Extraction

Extracting structured data from the Web

• Pattern or rule based extraction
• NLP extraction: apply linguistic analysis on free texts
• Shallow extraction: recover structures already on the Web
Hearst Patterns [Hearst, ACL 1992]

Succinct list of syntactic patterns expressing parent/child relationships, i.e., `instanceOf` or `isA`

Can you think of any pattern?

```
NP “such as” NP*
cities such as London, Paris, and Rome

“such” NP “as” NP* (“or” | “and”) NP
works by such authors as Herrick and Shakespeare

NP, NP* (“or” | “and”) other NP
bruises, wounds, broken bones or other injuries

NP “including” NP*
all common-law countries including Canada and England
```
KnowItAll [Etzioni et al., AI 2005]

First Web-scale application of Hearst Patterns

Important components

- Extractor: extracts entities/subclasses based on patterns
- Assessor: evaluate the extraction results
- Pattern Learner: learn additional patterns based on high quality entities
Extractor

For each pattern and each class

▪ Create a search engine query based on the pattern/class combination
  ▪ “companies such as”

▪ Pose the query to multiple search engines and collect all result documents

▪ Apply the pattern to those documents and extract tuples according to the rules

Analysis: Why companies such as Virgin Galactic and ...
www.houstonchronicle.com/.../Analysis-Why-companies-such-as-Virgin...
Oct 31, 2014 - Analysis: Why companies such as Virgin Galactic and Orbital take risks and endure losses. NASA, FAA should ask questions, but private, 'new ...
Extractor

Class: Company

Pattern: NP1 “such as” NPList2

Constraints: head(NP1) = “companies” & properNoun(head(each(NPList 2)))

Output: Company(head(each(NPList2)))

Keywords: “companies such as”
Leverage the notion of PMI: Pointwise Mutual Information

For each class, KnowItAll generates a *discriminator phrase* \( D \), usually just the name of the class, then:

\[
\text{PMI}(I, D) = \frac{\mid \text{Hits}(D + I) \mid}{\mid \text{Hits}(I) \mid}
\]

The higher the PMI value, the more likely the instance is a true instance

- i.e., the instance is always associated with the class
- \( \text{Hits}(D) \) can be ignored because it is the same across all instances
Rule Learner

Learn new rules based on extracted tuples of high confidence

- “paris and shanghai”

<table>
<thead>
<tr>
<th>Rule</th>
<th>Correct Extractions</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>the cities of &lt;city&gt;</td>
<td>5215</td>
<td>0.80</td>
</tr>
<tr>
<td>headquartered in &lt;city&gt;</td>
<td>4837</td>
<td>0.79</td>
</tr>
<tr>
<td>for the city of &lt;city&gt;</td>
<td>3138</td>
<td>0.79</td>
</tr>
<tr>
<td>in the movie &lt;film&gt;</td>
<td>1841</td>
<td>0.61</td>
</tr>
<tr>
<td>&lt;film&gt; the movie starring</td>
<td>957</td>
<td>0.64</td>
</tr>
<tr>
<td>movie review of &lt;film&gt;</td>
<td>860</td>
<td>0.64</td>
</tr>
<tr>
<td>and physicist &lt;scientist&gt;</td>
<td>89</td>
<td>0.61</td>
</tr>
<tr>
<td>physicist &lt;scientist&gt;,</td>
<td>87</td>
<td>0.59</td>
</tr>
<tr>
<td>&lt;scientist&gt;, a British scientist</td>
<td>77</td>
<td>0.65</td>
</tr>
</tbody>
</table>
Generate-and-Test Loop

1. extract new instances
2. find occurrences of seed instances
3. annotate text
4. generate new patterns

patterns
instances
Thought Exercise: Improving Recall

Any idea on how to improve recall?

List Extraction:

• Compose search queries that contain $k$ number of random instances from the class
• For each returned document, search for lists in the document
• Extract the unknown entries in the list as candidate instances

“shanghai paris london mumbai”

Rank those instances based on

• How many lists they appear in
• How good are the lists they appear in (i.e., how many known instances those lists contain)
Thought Exercise: Improving Precision

“health care companies such as hospitals, elderly care facilities, …”

Any idea for reducing the noise?

Requiring the body list containing at least a known entity

NP “such as” NP*
cities such as Bangkok, Nay Pyi Taw, and Ho Chi Minh City

The more known entities are required to present, the higher the precision, the lower the recall
Building isA Hierarchies

Iteratively apply Hearst patterns

[Kozareva and Hovy, EMNLP 2010]

“animals such as lion and ?” → {lion, tiger, jaguar}

“? such as lion and tiger” → feline

“? such as feline” → “Big Predatory Mammals”

“mammals such as felines and ?” → {felines, bears, wolves}
TextRunner [Banko et al, IJCAI 2007]

Extract all kinds of relations (beyond instanceOf or isA)

Main idea:

Using a deep linguistic labeler to generate training examples and train a classifier over a small set of relations, and leverage shallow features to estimate trustworthiness of relations for the entire Web.
**Self-Supervised Learner**

Deep linguistic parser to generate training examples to estimate trustworthiness of relations

- Linguistic parser: an NLP tool that converts a sentence into a syntax parse tree

Google is headquartered in Mountain View ➔

(Google, headquarter, Mountain View)
Self-Supervised Learner

Deep linguistic parser to generate training examples to estimate trustworthiness of relations

- Linguistic parser: an NLP tool that converts a sentence into a syntax parse tree
- An entity tagger that tags any base noun phrase (NP) as a potential entity
- For each pair of potential entities in the sentence
  - Collect possible relation labels (VBN)
  - Collect deep parse tree features: dependency, pronoun, etc.
- Label a triple \((e_1, \text{rel}, e_2)\) as a positive example if certain deep parse features are satisfied, label it as a negative example otherwise --- this is based on domain knowledge
Self-Supervised Learner

For each positive/negative example, collect shallow parse features

▪ Number of tokens between entities
▪ Number of stop words
▪ Etc.

Train a trustworthiness classifier based on those shallow features and a small set of examples

▪ Given a pair of entities and their label, produce a score as to whether their relation is likely to be true or not
Extractor and Assessor

Play similar roles as their counterparts in KnowItAll

The extractor extract candidate tuples

▪ Step 1: raw candidate generation, essentially using a simple part-of-speech tagger to generate pairs of entities
▪ Step 2: candidate classification using the classifier

The assessor ranks the remaining candidates using a probabilistic model based on the PMI model
Finding facts

Getting information (facts) about an entity:

- A fact is a pair regarding an entity: (attribute, value) about entity; or can consider a tuple (entity, attribute, value).

Ex: Chuck Norris

★ Height: 5’ 10”
★ Spouse(s): Gena O'Kelley, Dianne Holechek

(Chuck Norris, height, 5’ 10”)

(Chuck Norris, spouse, {Gena O'Kelley, Dianne Holechek})
Finding facts

Getting information (facts) about an entity:

Entity relationships / querying the graph

[who is chelsea clinton’s grandmother] =>

- (chelsea clinton, mother, A)
- (chelsea clinton, father, B)
- (A, mother, C)
- (C, mother, D)
- return {C, D}
Finding facts

Getting information (facts) about an entity:

Non-relationships (number values, other complex & organic values)

[when is chuck norris’ birthday] =>

(chuck norris, birthday, X) -> March 10, 1940

“March 10, 1940” is NOT an entity! It is an attribute on the Chuck Norris entity. (i.e. a payload of data on the entity).
Finding facts: extraction

Similar strategy as before, but use annotated text instead of entities:

- Annotate dates, time, date range (Ex: March 10, 2016)
- Annotate numbers (Ex: 2, 6, 299)
- Annotate measurements (Ex: 4 kg, 300 feet, etc.)

Discover phrases that represent same attribute:

Ex: (NYU) “founded” = “established”
Finding facts: importing

Take advantage of existing “databases”:

- Scrape sites like Wikipedia, Wikia, imdb, drugs.com, etc.
- Page structures like `<table>` or `<div>` hierarchies, repeated features to detect attributes and values
- ML what is a “quality” structure, and what is not:
Outline

Web-scale extraction
• Finding entities
• Finding relationships
• Finding facts

Search with knowledge
• Query tagging
• Query suggestion

Recent advancements
• Crowdsourcing knowledge
Improving Web Search via Query Enrichment

Goal: better query understanding by associating semantics with user queries

Case studies:

▪ Query tagging

▪ Query suggestion (using knowledge)
Studies on Query Tagging

Named entity recognition

- [Guo et al, SIGIR 2009]

Rich interpretation

- [Li, Wang, Acero, SIGIR 2009]
The Query Tagging Problem

Even simple named entity tagging is not easy

“first love lyrics”

➔ The real entity is “first love” of class “song”
Tagging Named Entities in Queries [Guo et al]

Challenges:

▪ Short text
▪ Fewer language features: e.g., no punctuation, no capitalization

Intuition:

▪ Use context to disambiguate
▪ Use query logs to learn probabilities
▪ Gather class labels on seed entities
Probabilistic Model

Model the tagging problem as computing the probabilities of all possible triples, \( (e, t, c) \), that can represent the query:

- \( e \): entity
- \( t \): context
- \( c \): class label for the entity

“first love lyrics”

- (“first love”, “# lyrics”, song), or
- (“first”, “# love lyrics”, album), or
- (“love” “first # lyrics”, emotions), or
- (“lyrics”, “first love #”, music)

The one with the highest probability can be considered as the correct tagging.
Probabilistic Model

The learning problem is:

\[
\max_{i=1}^{N} \prod \Pr(e_i, t_i, c_i)
\]

\[\Pr(e, t, c) \text{ can be estimated as:}\]

\[\Pr(e, t, c) = \Pr(e) \Pr(c|e) \Pr(t|e, c) = \Pr(e) \Pr(c|e) \Pr(t|c)\]

- I.e., “lyrics” is more likely associated with songs, regardless which song
Training and Prediction

Step 1: Gather seed set of (entity, class) pairs

Step 2: Match the seed set with query log and gather their contexts: \((e_{seed}, t)\)

Step 3: Use the contexts gathered from step 2, match with the query log again and gather expanded entities: \((e_{expanded}, t)\)

Conditional probability estimates:

- \(Pr(e)\): occurrence frequency in logs
- \(Pr(t|c)\): learned from Step 2.
- \(Pr(c|e)\): learned from Step 3 with fixed \(Pr(t|c)\) from Step 2.

Prediction: apply to all possible query segmentations
Results

12 Million unique queries ➔ tagged 0.15 Million
- Recall is quite low

Sampled 400 queries for evaluation
- 111 movie, 108 game, 82 book, 99 song
Studies on Query Suggestion

Query to Query

▪ [Szpektor, Gionis, Maarek, WWW 2011]

Entity to Query

▪ [Bordino et al, WSDM 2013]
The Query Suggestion Problem

Enormously popular with the users
- Works very well for head queries

Approaches
- Query similarity
  - E.g., Cosine similarity, edit distance
- Query flow graph
  - Leveraging co-occurrences of queries in the same query session
The Query Suggestion Problem

More challenging for long tail queries

Query similarity often lead to wrong suggestions

By definition, they are very rare in query log

Proposed approaches:
- Dropping non-critical terms [Jain, Ozertem, Velipasaoglu, SIGIR 2011]
- More interestingly, query templates!
Query Flow Graph [Boldi et al, CIKM 2008]

Constructed from query session logs

Nodes are queries

Create an edge \((q_1, q_2)\) if:
- \(q_2\) appeared as a reformulation of \(q_1\) in a session

Edge weights can be assigned in many ways
- \(Pr(q_2|q_1) = f(q_1, q_2) / f(q_1)\)
- \(PMI(q_1, q_2) = \log(f(q_1,q_2) / f(q_1)f(q_2))\)
Query Template Flow Graph [Szpektor et al]

Intuition:

▪ If, users often search “new york restaurants” after searching for “new york hotels” and similarly for other popular cities such as “shanghai”, “paris”, etc.

▪ Then, “pkoytong restaurants” can be a good recommendation candidate for “pkoytong hotels” since “pkoytong” is a also a city
Query to Template Edges

Query entity tagging techniques made query template generation possible!

- “new york restaurants” ➔ “<city> restaurants”
- Essentially, knowledge can be used to enrich the query to address many issues associated with long tail queries

Computing edge weights between query and template

- Assuming a hierarchical ontology
- $S(q, t)$ is computed based on where in the hierarchy the query entity in $q$ is matched

\[ S_{qt}(q, t) = \alpha^{d(z,e)} \]
Template-to-Template Edges

Creating edges between templates

- A template-to-template edge occurs if and only if a query-to-query edge occurs and the two queries match the two templates, respectively, with the same entity
  - nyc restaurants ➔ <city> restaurants; nyc hotels ➔ <city> hotels
  - <city> restaurants ➔ <city> hotels

Computing edge weights between templates

- The more supporting query-to-query pairs there are, the higher the weights

\[
S_t(t_1, t_2) = \sum_{(q_1, q_2) \in \text{Sup}(t_1, t_2)} s_{qq}(q_1, q_2),
\]

\[
s_{tt}(t_1, t_2) = \frac{S_t(t_1, t_2)}{\sum_t S_t(t_1, t)}.
\]
Generating Recommendations

Let $S(x, y)$ be the probability of reaching $y$ from $x$ in the graph

- E.g., product of all the edge weights on the path from $x$ to $y$.

The score of $r(q_1, q_2)$ can be computed as

- $S(q_1, q_2)$ based on original query flow graph, plus
- $\sum_{ij} ((q_1, t_i), S(t_i, t_j), S(q_2, t_j))$ based on the query template flow graph

Tough cases remain

- E.g., entities that do not appear in the ontology hierarchy, which is much more common in long tail queries
Results

The query template graph: 95M queries, 60 candidate templates per query

- Number of edges is linear to the number of nodes

<table>
<thead>
<tr>
<th></th>
<th>QFG</th>
<th>QTGF</th>
<th>relative increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>total in test-set</td>
<td>3134388</td>
<td>3134388</td>
<td></td>
</tr>
<tr>
<td>upper-bound coverage</td>
<td>(22.65%)</td>
<td>709832</td>
<td>(28.17%)</td>
</tr>
<tr>
<td># in top-100</td>
<td>(16.97%)</td>
<td>531854</td>
<td>(25.49%)</td>
</tr>
<tr>
<td># in top-10</td>
<td>(9.49%)</td>
<td>297462</td>
<td>(20.74%)</td>
</tr>
<tr>
<td># ranked highest</td>
<td>(2.86%)</td>
<td>89740</td>
<td>(10.01%)</td>
</tr>
<tr>
<td>MAP</td>
<td>0.050</td>
<td></td>
<td>0.137</td>
</tr>
<tr>
<td>avg. position</td>
<td>18.35</td>
<td></td>
<td>8.3</td>
</tr>
</tbody>
</table>
Entity Query Graph [Bordino et al]

Recommending queries when a user shows interests in an entity, e.g.:

▪ When a user is visiting a Wikipedia page
▪ When a user searches for an entity
▪ When a user’s profile has an entity

Similar idea to extend query flow graph, but using entities instead of templates

Again, enabled by query tagging techniques
Entity Query Graph

Entity-to-query edges

\[ w_{\mathcal{E}Q}(e \rightarrow q) = \frac{f(q)}{\sum_{q_i \mid e \in \mathcal{E}(q_i)} f(q_i)} \]
## Results

Generate recommendations based on personalized PageRank over the Entity Query Graph

Data: 200M queries; 100M entities; linear number of edges

<table>
<thead>
<tr>
<th>Testset</th>
<th>Label</th>
<th>EQGraph</th>
<th>Reverse IR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wikipedia pages</td>
<td>Related and interesting</td>
<td>62.7%</td>
<td>33%</td>
</tr>
<tr>
<td></td>
<td>Related but obvious</td>
<td>3.3%</td>
<td>41.5%</td>
</tr>
<tr>
<td></td>
<td>Unrelated</td>
<td>34%</td>
<td>25.5%</td>
</tr>
<tr>
<td>Yahoo! News + Yahoo! Finance</td>
<td>Related and interesting</td>
<td>52%</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td>Related but obvious</td>
<td>2.3%</td>
<td>34.3%</td>
</tr>
<tr>
<td></td>
<td>Unrelated</td>
<td>45.7%</td>
<td>25.7%</td>
</tr>
<tr>
<td>Full testset</td>
<td>Related and interesting</td>
<td>58%</td>
<td>36.1%</td>
</tr>
<tr>
<td></td>
<td>Related but obvious</td>
<td>2.9%</td>
<td>38.4%</td>
</tr>
<tr>
<td></td>
<td>Unrelated</td>
<td>39.1%</td>
<td>25.5%</td>
</tr>
</tbody>
</table>
Outline

Web-scale extraction
• Finding entities
• Finding relationships

Search with knowledge
• Query enrichment
• Query suggestion

Recent advancements
• Crowdsourcing knowledge
Crowd Sourcing: Robots vs Humans
Why Now? (video clip)

The Web enabled people to connect virtually and brought in two main advantages that were previously impossible.

**Scalability**
- 10 users vs 10,000 workers

**Cost effectiveness**
- $10+/hour vs $0.01/task

It is now possible to design a task and have thousands of people working on it cheaply at a moment’s notice!
Reading Material


Play Games with a Purpose

Some AI problems that are extremely difficult for current computer programs to do.

Example, Image labeling and object recognition

• Require many hours of human work to generate training examples.
• But those “human work” are really simple for human to do.

And people like to play games …
9 BILLION HUMAN-HOURS OF SOLITAIRE WERE PLAYED IN 2003

EMPIRE STATE BUILDING
7 MILLION HUMAN-HOURS
(6.8 HOURS OF SOLITAIRE)

PANAMA CANAL
20 MILLION HUMAN-HOURS
(LESS THAN A DAY OF SOLITAIRE)

slide borrowed from Luis von Ahn
Play Games with a Purpose

Some AI problems that are extremely difficult for current computer programs to do.

Example, Image labeling and object recognition

• Require many hours of human work to generate training examples.
• But those “human work” are really simple for human to do.

And people like to play games …

Can we design games so that people can play the game while helping some good cause?
The ESP Game

GUESSING: **CAR**

GUESSING: **HAT**

GUESSING: **KID**

SUCCESS!
YOU AGREE ON **CAR**

PLAYER 1

GUESSING: **BOY**

GUESSING: **CAR**

SUCCESS!
YOU AGREE ON **CAR**

PLAYER 2
HINT

Noun

BUSH
Basic Rules of Play

*Incentive*: each session has a limited time and the goal is to get as many points as possible.

Each boom reveals a small portion of the image and takes time. Therefore, users want to use as few boom as possible and reveal only the relevant portion of the image.

Alternatively, the score can be a combination of time and image area revealed.
Utility of Peekaboom

Images and labels come from the results of the ESP game!

How the words relate to the image
- Verb, noun, etc.

Bounding boxes of the objects represented by the word

Salience detection: boom usually happens on the part of the image that is most salient to the word

Elimination of poor image-word pairs
Combating Cheating / Collusion

Player’s Queue
- Start a game puts the player on a queue
- Pairing happens periodically or when there are enough players on the queue

Seed Images
- Images with verified metadata to detect players who are consistently booming at the wrong places and guess the correct words

Prevent miscreant communications via guesses:
- guess: hi, my gtalk is
- guess: cheater_on.peekaboom
- guess: give me the word
- guess: and I will enter it
Human Computation vs Crowd Sourcing

Human Computation

- A more scientific concept, opposite to machine computation
- Leverage human intelligence to solve hard AI problems
  - Users get rewarded through enjoyment
- E.g., Peekaboom

Crowdsourcing

- A business concept, evolved from outsourcing
- Open call for workers through open platforms
- Workers are paid cheaply or even work voluntary
- Good solutions often percolate up, through the wisdom of crowds
- E.g., Wikipedia
Challenges of Crowdsourcing

How to recruit (and retain) users
- Pay for work: e.g., AMT
- Pay for result: e.g., crash the super bowl
- Ask for Volunteers: e.g., Wikipedia
- Implicitly: e.g., CAPTCHA

How to define and budget the tasks
- Simple: label an image
- Complex: find information about an entity, use some application and complete a survey
- Creative: produce a 30-minute video
- Impactful vs marginal: creating an entity in a knowledge base versus create a rule that can be used to find many entities
Challenges of Crowdsourcing

How to consolidate user contributions

- Simple aggregate: majority rules
- Complex integration: involving deep integration, conflict resolution, etc.
- Crowd driven: the community itself votes on the best solution
- No consolidation: Wikipedia, where the last editor decides

How to perform quality control

- Natural variations are common for human produced artifacts
  - Given an image of the IBM logo, labels can vary from IBM to I.B.M. to International Business Machines, etc.
- Malicious users
  - Detection: similar to those employed by Peekaboom
  - Monitoring: Wikipedia Arbitration Committee
Especially Challenging: Long Tail

Key challenges

• Finding knowledgeable users is extremely hard for long tail topics that are not familiar to many people
• Making those people work for money is even harder
  • Domain experts who are educated or otherwise busy

Key insights

• Using advertising campaign to recruit users
• Using games to keep users engaged

Quizz: acquiring knowledge about long tail topics using crowdsourcing mechanisms that are difficult to obtain using automated means.
Overview of the Quizz System

- **Advertising (Section 2)**
  - Internet Users (display ads)
  - Internet Users (sponsored search ads)

- **Measuring User Contributions (Section 3)**
- **User Contribution Measurement**
- **Feedback on conversion and contributions for each user click**

- **Feedback and Incentives (Section 5)**
  - Correct Answers: 30633
  - Correct (%): 48%

- **Exploration vs Exploitation (Section 4)**

- **Question**
  - Serve Calibration or Collection Question?
    - Calibration Questions (with known answers)
    - Collection Questions (with uncertain answers)
Ad Campaign to Recruit Users

Users are not paid, they are simply interested in the topic

• The advertising campaign does cost money

Using disease symptoms as an example topic

• Patients
• Doctors!
Campaign Optimization

Two important aspects about user who click through

• Enjoying the quizz and thus happy to complete them
• Competence in answering the quizz questions correctly

The Ad Campaign needs to optimize for attracting users who are both enjoying the quizz and competent.
The Quizz

Calibration questions
• Estimate user quality

Collection questions
• Where real utilities are

Using information gain to measure the value of a user session
• Higher with better quality
• Higher with more volume
User Participation

Conversion rate steadily improved with continuous optimization, 20% to 50%+
User Participation

Conversion rate steadily improved with continuous optimization, 20% to 50%+
### Cost of Acquisition

<table>
<thead>
<tr>
<th>Quiz</th>
<th>@99%</th>
<th>@95%</th>
<th>@90%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disease Causes</td>
<td>$0.07</td>
<td>$0.05</td>
<td>$0.04</td>
</tr>
<tr>
<td>Disease Symptoms</td>
<td>$0.02</td>
<td>$0.01</td>
<td>$0.01</td>
</tr>
<tr>
<td>Treatment Side Effects</td>
<td>$0.13</td>
<td>$0.10</td>
<td>$0.07</td>
</tr>
<tr>
<td>Artist and Albums</td>
<td>$0.16</td>
<td>$0.13</td>
<td>$0.09</td>
</tr>
<tr>
<td>Latest Album</td>
<td>$0.09</td>
<td>$0.07</td>
<td>$0.05</td>
</tr>
<tr>
<td>Artist and Song</td>
<td>$0.54</td>
<td>$0.42</td>
<td>$0.31</td>
</tr>
<tr>
<td>Film Directors</td>
<td>$0.07</td>
<td>$0.05</td>
<td>$0.04</td>
</tr>
<tr>
<td>Movie Actors</td>
<td>$0.22</td>
<td>$0.18</td>
<td>$0.13</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>$0.16</strong></td>
<td><strong>$0.12</strong></td>
<td><strong>$0.09</strong></td>
</tr>
</tbody>
</table>
Quality of Answers

Unpaid users vs. hourly (oDesk) vs. piecemeal (MTurk)

% of correct answers vs. Log2(Submitted Answers)

Source
- MTurk
- oDesk
- Quizz
Summary

Lots of work are happening in knowledge search

Lots of challenges remain:

- Knowledge base construction and maintenance
  - Automated
  - Crowdsourcing
- Search beyond simple entities
  - Q/A and NLP search
- Integrating many approaches into one (or multiple?) comprehensive graph(s)