Improving Search Engines Using Human Computation Games

Hao Ma *
Dept. of CSE
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
hma@cse.cuhk.edu.hk

Raman Chandrasekar, Chris Quirk
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
{ramanc,chrisq}@microsoft.com

Abhishek Gupta *
Digital Media, LCC
Georgia Institute of Technology
Atlanta, GA 30332, USA
abhishek.gupta@gatech.edu

ABSTRACT
Work on evaluating and improving the relevance of web search engines typically use human relevance judgments or clickthrough data. Both these methods look at the problem of learning the mapping from queries to web pages. In this paper, we identify some issues with this approach, and suggest an alternative approach, namely, learning a mapping from web pages to queries. In particular, we use human computation games to elicit data about web pages from players that can be used to improve search. We describe three human computation games that we developed, with a focus on Page Hunt, a single-player game. We describe experiments we conducted with several hundred game players, highlight some interesting aspects of the data obtained and define the 'findability' metric. We also show how we automatically extract query alterations for use in query refinement using techniques from bitext matching. The data that we elicit from players has several other applications including providing metadata for pages and identifying ranking issues.

Categories and Subject Descriptors
H.3.m [Information Retrieval]: Miscellaneous; H.5.3 [HCI]: Web-based Interaction

General Terms
Measurement, Design, Experimentation, Human Factors.

Keywords
Web Search, Human Computation Games, Findability, Query Alterations, Relevance

1. INTRODUCTION
Web search engines have become an integral part of our everyday lives. It is clearly important to evaluate the "goodness" of search engines to help improve the search experience of users and advertisers, and build traffic and revenue. Web search today typically retrieves several documents relevant to queries issued by users. Hence evaluation methods are usually based on measuring the relevance of documents to queries. One common method of evaluating relevance is to use large hand-annotated evaluation corpora where query/document pairs are assigned relevance judgments. Another method is to use implicit measures of relevance, such as identifying clicks on results. Although the click-based method is subject to noisy data issues, the amount of data available is potentially limitless. Compared to the hundreds of queries used in typical TREC collections, the vast quantities of clickthrough data available with search engines can support a much more extensive evaluation.

Both these evaluation methods (human relevance methods and click log methods) are based on pages surfaced by a search engine. However, some pages may never get surfaced for a variety of reasons, including bad ranking, indexing problems, network issues, improper deduplication of URLs, improper spam detection, etc. If a page is not surfaced (or is surfaced with insufficiently high rank), it will not enter the pool of results to be evaluated; nor will it figure in clickthrough data since no one can click on it.

To avoid problems in finding pages relevant to a given query, we suggest an alternative approach, looking in the other direction: given a web page, find the queries that will effectively surface this web page. This can potentially provide us with new insights into the relevance pipeline. If starting with queries focuses on how effectively search engines can retrieve documents relevant to specific queries, starting with web pages focuses on the findability of the pages indexed by the search engines. However, it would be time-consuming, expensive and infeasible to study every web page, and somehow obtain queries to find these web pages. We propose instead to use games to collect relevant data.

1.1 Using Games to Garner Data
There has been a lot of interest in using games to transform businesses, helping them become more competitive and profitable. Edery and Mollick [4] provide an excellent overview of this landscape. Chapter 5 of McDonald et al. [12] describes how productivity games have been used in software testing.

Recently, the idea of employing human computation to solve research tasks was popularized with the introduction of the ESP game [18]. As reported in [19], more than 200 million hours are spent each day playing computer and video games in the U.S. Indeed, by age 21, it is said that the average American has spent more than 10,000 hours playing such games. The idea behind human computation games is to harness the time and energy spent

*This work was done when the authors were on summer internships at Microsoft Research.

CIKM'09, November 2–6, 2009, Hong Kong, China.

on playing games to solve computational problems that are otherwise difficult to tackle.

The ESP game, developed by von Ahn et al. [18], brings web users together to participate in a large-scale image labeling problem in the guise of a guessing game. Other 'Games with a Purpose' [17, 19] have been developed to locate objects in images [23], collect common-sense facts [22], improve accessibility of the Web [21] and classify the query intent of users [8]. Similar games have been developed to tag images [1], and collect data for automated directory assistance [14]. Such games have become so popular that there are now toolkits, such as the HCToolkit from Law et al. [7], to simplify and automate the process of making a human computation game.

von Ahn and Dabbish [19] categorize human computation games into output-agreement games, inversion-problem games, and input-agreement games. In output-agreement games (e.g. the ESP game) two players are given the same input and must produce outputs based on the input; to win the game, both players must produce the same output. The inversion-problem games are a generalization of Peekaboom [23], Phetch [20] and Verbosity [22]. In these games, in each round, one player is assigned to be the "describer", and the other player is assigned to be the "guesser". The describer produces outputs that are sent to the guesser based on the input. The winning condition is that the guesser produces the input that was originally given to the describer. Finally, the input-agreement games represent a generalization of games like TagATune [9]. In each round, both the players are given inputs that are known by the game to be the same or different; but the players do not know if they are same or different. The players are instructed to produce outputs describing their input. Both players win if they both correctly determine whether they have been given the same or different inputs.

Whenever we gather data from the web, we must be concerned with quality control. The games described above typically involve two players who collaborate to find 'true' answers for each task; truth is determined by their agreement on results. To prevent collusion and data corruption issues, players are matched randomly. When there are an odd number of players, these games typically back-off to using data from previously recorded (two-person) games to simulate one of the players. This may degrade the gaming experience and thus the data quality. Also, these collaborative games tend to elicit tags/descriptions that are very general, since it is easier to agree on generalities than specifics. Partial remedies include the use of taboo words and specificity scores to encourage more specific labels, but this can still be a problem. Weber et al. [24] recently showed that the labels in ESP Game can be deduced from other labels present using language modeling. They developed a robot which played the ESP game, *and with no information from the images*, agreed with a randomly assigned partner 69% of the time on all images, and 81% when they had some off-limit (taboo) words assigned to them. Weber et al. have a range of palliative suggestions, including changing scoring, changing the timing mechanism, motivating players, hiding taboo terms etc.

In this paper, we seek to develop an efficient and effective human computation model for improving web search. We avoid the issues mentioned above primarily by using a single-player model. We focus primarily on Page Hunt, a single-player game where the player attempts to devise a query that will surface a given page in the top N results from a search engine; thus 'truth' is determined by the search engine used in the game.

The data that we elicit from players has several applications including providing metadata for pages, providing query alterations for use in query refinement, and identifying ranking issues. We describe an experiment we conducted with over 10,000 game players, and highlight some interesting aspects of the data obtained. We define a new metric we call 'findability', and describe one application for the data we collect – that of automatically extracting query alterations using the technique of bitext matching.

1.2 Research Questions and Issues

The overarching research question that we tackle is: Can we use human computation games to get useful data and develop an end-to-end process to improve web search? This bigger question can be broken into the following questions:

- Games have to be (by definition) fun to play and so in turn, we need to ask: Can this be made a fun game?
- Is the data we get from Page Hunt comparable to data we get from other sources? Does the nature of the game change the data?
- Finally, how do we get useful data from this? Can we define a process that extracts useful information from the data?

2. PAGE HUNT DESIGN

Once we decided to use human computation games, we came up with a set of design goals. We wanted a simple, fun game that people want to play (rather than be forced to play!). We wanted it to have a great user experience, be visually appealing, desirably be whimsical and definitely not be staid. We wanted it to be a web experience (rather than a downloaded game), with little or no server load. We wanted the game to be simple to understand and describe. We decided to use time limits to encourage quick play. Our goal was to design our game to avoid any bias in data collection.

As we started work, we designed and developed three related games, which we describe in this section. However, as mentioned earlier, in this paper we will focus primarily on Page Hunt, a single-player game where players hunts down given web pages with their queries. We will briefly describe the other two games that we developed: Page Race and Page Match.

2.1 Page Hunt

The basic idea of Page Hunt is to show the player a random web page, and get the player to come up with a query that would bring up this page in the top few results (say in the top 5) on a search engine. Fig. 1 shows a screenshot of this game. The web page being 'hunted' is shown in the background. The player types in queries and looks at results returned in the floating operating panel, currently in the lower right corner. This panel turns almost transparent when not in focus. The border of the panel relays important feedback information through animation and color changes. Game play goes thus:

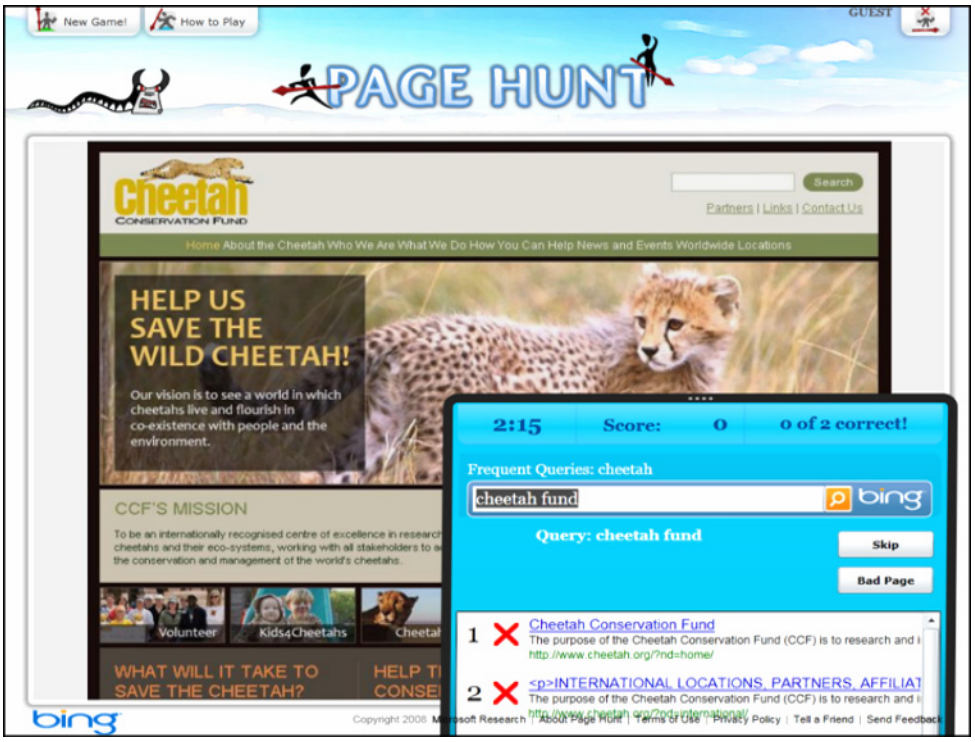

Figure 1. A screen shot of the Page Hunt game, just after an unsuccessful 'hunt'. The web page being hunted is in the background, and the operating panel with the user's query and search results floats on top.

1. The player gets a random web page with URL U. The player can see just the Web page, but not the URL of the page.

2. The player views the page, and types in a word (or a phrase) as the current query Q.

3. The system retrieves the top N search results for this query from a search engine and displays them. For each result, the rank, title, description or snippet and the URL are displayed. Against each result, a big check mark is shown if the match is successful, and a big cross is displayed if it is not.

We can use any web search engine to get these results. In Page Hunt, we get results from the Bing search engine (www.bing.com) using their public SOAP API.

4. This match is successful if the URL U is in the top N results for a given query Q. Players get points based on the rank of U in the result set: 100 points for rank 1, 90 for position 2 etc. Thus the search engine in the background determines if the player wins or loses.

We check for exact or near-exact URL matching, with some normalization (e.g. to handle http://a.com pointing usually to the same site as http://a.com/default.htm or http://a.com/default.aspx). But we do not attempt any sort of content matching to identify matching URLs.

5. If query Q does not lead to success, the player edits the query to change the phrase or add a word/phrase. If the query leads to success, the player's score gets updated, and the game advances to the next web page.

6. This is repeated for each page till the player quits or hits a fixed time limit for the game.

7. If players get stuck on a page, they can skip to the next page with no penalty; we do not force the player to provide tags for each page. If they see a page which is rendered badly or is bad in any way, they can mark it as bad, again with no penalty. If several players tag a page as bad, the page is reviewed and removed if appropriate.

At each step, we record the following: the player's screen name (not connected to the player's real identity in any way; the game can also be played anonymously), the web page id, the query that was tried, whether it was right and if so at what rank position, the time, and the points the player got for this query.

The terms (queries) that the player provides at each step acts as a tag or label for the page. When the player gets it right, the label is valuable; but even when it is wrong, the label provided could be useful.

At the end of each game, the player gets the option to review each page, the queries tried and the results obtained. This can be valuable in training people how to query better.

We want players to use appropriate query terms to get to the given page; we do not want random sequences of words from the page used as queries. For this reason, a transparent overlay on the web page prevents players from cutting and pasting long phrases from the page as queries. Players may still type in long phrases from the page, but since search engines typically have a query length limit, players soon learn to be discriminating.

The quality of the game depends on the web pages displayed. Although search engine companies have abundant clickthrough data, not all the web pages have enough adequate metadata associated with them. The web pages we use in Page Hunt are those that have been identified by a search engine as requiring special treatment. Some are pages with very little text, or pages which search engines tend not to surface; other pages are about celebrities, weather etc.

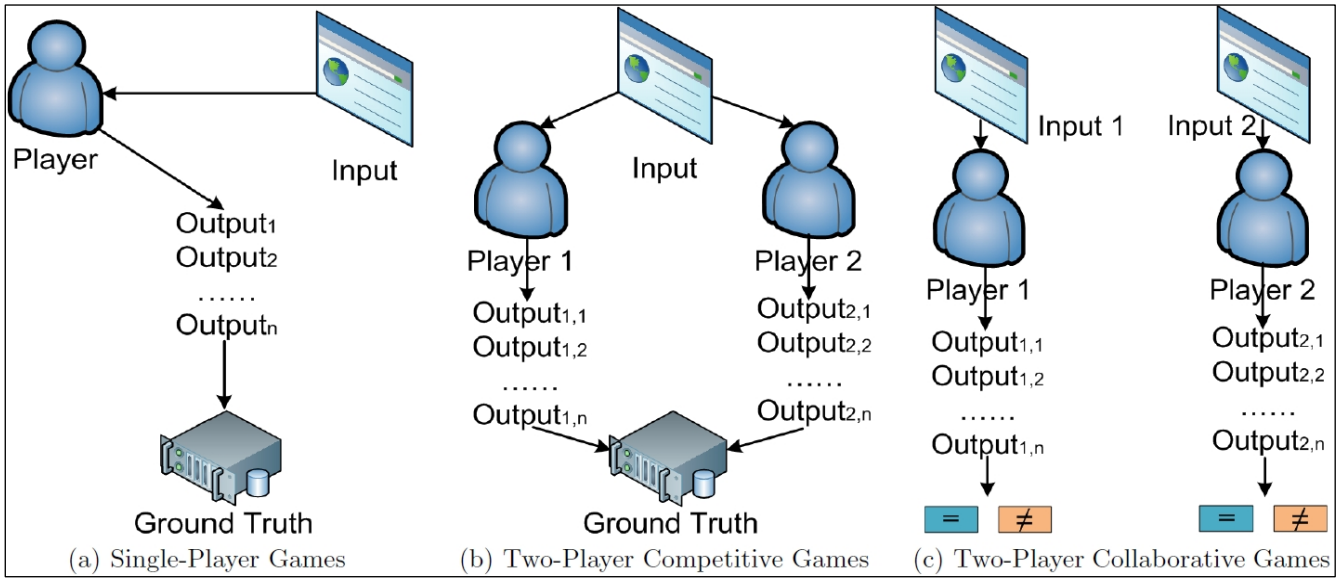

Figure 2. Generalization of Single-Player Games, Two-Player Competitive Games and Two-Player Input-Agreement Collaborative Games. In Fig. 2(a), the player wins if one of the outputs matches the ground truth, where truth is determined by a program. In Fig. 2(b), the player whose output matches the (program-defined) ground truth first wins. In Fig. 2(c), both players win if both players correctly determine and agree whether they have been given the same or different inputs.

As we will see later, not all pages are easy to 'hunt down' – some are easier than others. It is easy to adapt the game to the player's skill level with an appropriate choice of web pages.

Single player human computation games such as Page Hunt are generalized in Fig. 2(a).

2.1.1 Making Page Hunt Fun

Based on our game design goals, we chose to implement Page Hunt using Microsoft Silverlight, since it gave us a good platform for development, access to animation and interactivity, and browser independence.

Page Hunt also includes several features to increase fun in game playing, such as timed response, score keeping, top-players list, frequent queries (a concept similar to taboo queries), and occasional bonus points. Because player engagement is important in this context, we provide additional details on these points.

2.1.1.1 Timed Response

Setting limits on game sessions introduces challenge into a game in the form of a timed response [10, 11]. In Page Hunt, we set the time limit for each round as 3 minutes. The time limit and time remaining are displayed throughout the game, pushing players to keep trying queries quickly, one after the other.

2.1.1.2 Score Keeping and a Leader board

Page Hunt provides a simple score based on the rank of the URL U in the result set. Top players who login into the game can see their names on the leader board.

2.1.1.3 Frequent queries

We do not have the concept of taboo queries in Page Hunt, where specific terms are completely disallowed. However, for a small fraction of web pages displayed, we show frequent queries associated with the page. If players get stuck on such a page, they can issue these frequent queries. However, they get bonus points if they avoid using these frequent terms. This encourages forward movement in the game and prevents players from getting frustrated with the game.

2.1.1.4 Randomness

In Page Hunt, the randomness comes out in two of its features: (1) the web pages displayed are randomly selected for each player, and (2) during scoring, the player's winning score is randomly doubled for a page with a probability of 10% and randomly tripled with a probability of 5%. These random bonuses increase the fun factor of the game, and act as an incentive to play again.

2.2 Page Race and Page Match

Page Race, a two-player competitive game, is a natural extension of Page Hunt. Fig. 3 shows the operating panel of this game. In this game, a player is randomly matched with another. Both players are shown the same random web page. As in Page Hunt, the players type in a word (or a phrase) as the current query after viewing the content of this web page. The player who first brings up the web page within the top N search results wins the points. In this game, the players review the last query issued by them and the related search results; in addition, they can see the last query issued by their competitor and the related search results. Each player can thus learn from his/her competitors' queries and related search results.

Page Match is a collaborative game (see Fig. 3 for a screenshot) where two players are randomly paired up. Both players are shown a web page each. These pages are the same with a 50% probability, and different but related with a 50% probability. The players type in a word (or a phrase) as the current query Q after viewing the content of their web page. The players then compare

Figure 3. Screen shots of the operating panels of the Page Race (on the left) and Page Match (on the right) games.

their search results with the search results generated by their partners, and then assess whether their web pages are the same or different. If both players correctly determine *and agree* whether the pages they have been given are the same or different, both players win points.

The Page Match and Page Race games were built with an early version of the HCToolKit [7], and the games share some code components.

Fig. 2(b) and Fig. 2(c) depict generalized versions of the Page Race and Page Match games.

3. EXPERIMENT

As mentioned above, the focus of our experiment is the Page Hunt game. We released the game to the web, we conducted a pilot study with over 10,000 web users. On the game page, we told the users about the game, described the rules, and gave them a web link to try the game. We seeded the system with 698 URLs for which better labels were required. We did not offer any inducements (gifts or payments) since we did not want these inducements to bias their play. Thus participation was completely voluntary.

In the next section, we present evidence showing that Page Hunt was fun to play and that people provide valuable research data while playing.

4. DATA ANALYSIS AND DISCUSSION

In this section, we consider the research issues listed in Section 1.2, and provide answers to each of them.

4.1 Was the Game Fun?

During this experiment, over 10,000 people on the web played Page Hunt over a period of ten days, generating over 123,000 non-null labels on the 698 URLs in the system. On average every player contributed over 12 labels, and every URL has over 170 labels.

Players sent comments like "addictive", "I love the app. It has a sticky, viral nature to it", "Fun game!" and "…It's a great game. Passing it along …". The players who were at the top of the leader board kept playing to keep their names on the leader board.

Judging from these numbers, and from the comments we got, this game seems to be fun.

4.2 Nature of Queries Elicited

In an experiment of this nature, it is important to check if the queries we elicit from the players are comparable to the queries that they would have posed to a standard web search engine. From the pilot data, we took a random sample of successful 100 <query, web page URL> pairs. Two of us analyzed these 100 pairs and categorized the queries as OK, Over-specified or Under-specified. OK queries are just what one needs to get the associated web page. An over-specified query is one like [start here medline plus urticaria] for the web page http://www.nlm.nih.gov/medlineplus/hives.html. The query without the words 'start here' is enough to get this page in the top 5 results. Similarly, since we would need more query words than [nelson county] to get to the specific page

http://www.nelsoncounty.com/directory/discuss/msgReader$340. So the query [nelson county] is treated as an under-specified query for this page.

In our analysis, 78% of the queries were OK, 15% were over-specified and 7% were under-specified.

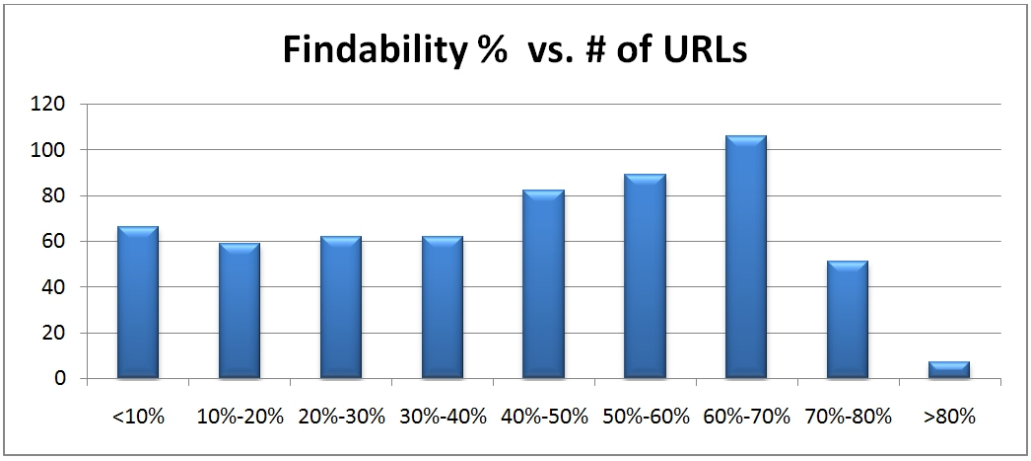

Figure 4. URL Findability Levels. X-axis is findability, and Y-axis is the number of URLs.

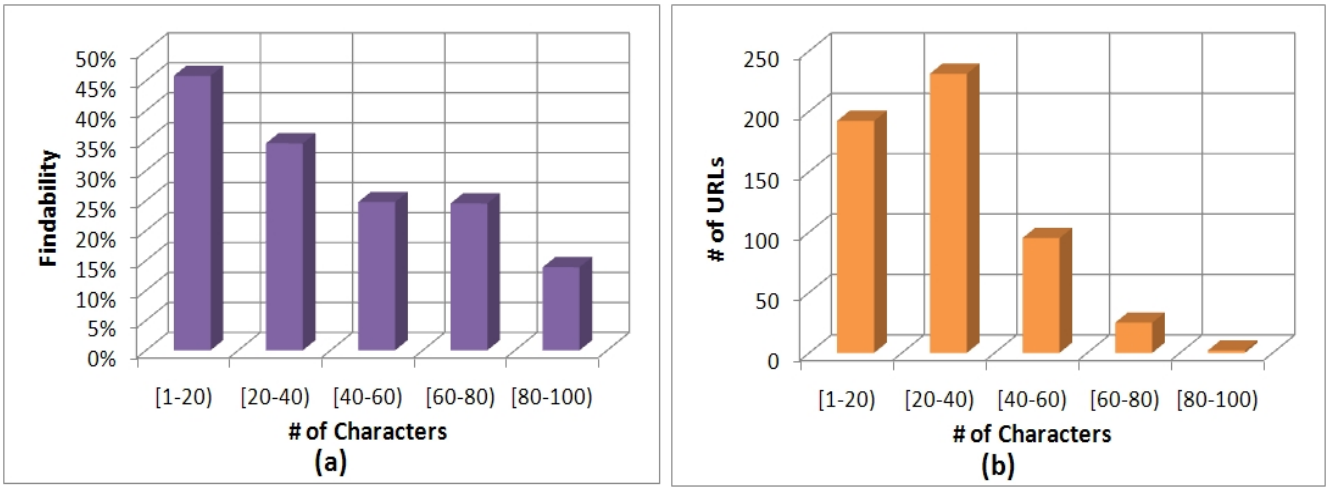

Figure 5. Findability Analysis of URLs

Because the web pages being 'hunted' are in front of the players as they create their queries, we expected the queries to be more like 'known item' searches than exploratory searches. We expected queries to be over-specified in general. But, contrary to our expectations, the queries are predominantly (78%) similar to queries given to web search engines, while only slightly above a fifth were over-specified or under-specified.

4.3 URL Findability

4.3.1 Findability Levels

From the data collected, we find that some of the URLs are easily found (or 'hunted' down), while some others are difficult to get in the top search results. Fig. 4 shows the findability distribution of the URLs. The X-axis of this figure shows findability ranges of URLs. A 100% findability level indicates that this URL can be easily located, since in every instance where a player was presented with this URL, the player was able to successfully identify a query that brought up this URL to the top 5 search results. On the other hand, if a URL has a 0% findability level, then that indicates that in no instance where a player was presented with this pagfe was the player able to identify a query that successfully brought up this URL to the top 5 search results.

From Fig. 4, we can see that about 10.0% of the 698 URLs in our database have >70% findability while roughly the same (11.3%) have <10% findability.

4.3.2 Findability as a Function of URL Length

As a first step in investigating the factors that impact the findability levels of URLs, we evaluate the impact of the length of URLs. In Fig. 5(a) and Fig. 5(b), we show how the length of a URL impacts the URL findability levels. Here the length of a URL is measured as the number of characters it contains, not including 'http://'.

From Fig. 5(a), we draw the conclusion that as the length of URL links increases, the URLs are harder to locate through search engines. Fig. 5(b) shows the distribution of the number of characters in each URL.

Note that findability could have other uses. If we ran a game like Page Hunt continuously, assuming a reasonably large set of web pages, and a random, changing set of players, the average findability of the pages could provide valuable insights into search engine accuracy. The findability metric can be used to evaluate the overall and comparative goodness of search engines.

4.4 Query Alterations from Game Data

On most search engines, if you search for, say, [Wash. DC], the query is internally modified as if it were equivalent to a search on [Wash. DC] ORed with [Washington, DC]. This idea of altering the user's query using other terms which may have been intended is called query alteration. The query alterations that are used in a search engine may be obtained using a variety of sources. In this section, we describe one way to learn query alterations from Page Hunt game data.

Consider a web page like http://www.labor.state.ny.us/ . When we tried this on our pilot players, we got a number of queries that successfully retrieved this page, some of which are shown in Table 1.

It is easy to see that the query NYSdol should have "new york state department of labor" as a possible alteration, that "department" should have "dept" as an alteration etc. But how can we automatically extract these alterations?

Table 1. Queries for the NY State Dept of Labor page

New York department labor
New York state department of labor
New York state dept of labor
NY Department Labor
NY labor
NYS dept of labor
NYSdol

We use the intuition that different queries for the same page can be viewed as different paraphrases in the same language, or translations of the same concept in different languages. There has been work in statistical machine translation (MT) to learn such paraphrases or matching segments across document pairs (hence "bitext matching"). We use bitext matching, as described below, to learn alterations from our game data.

4.4.1 Bitext Matching

In recent years, statistical approaches to machine translation that learn translational equivalents from bitexts have shown great promise as effective translation techniques. Modern statistical methods learn translational equivalents directly from parallel data with little to no outside supervision; in many cases, only a tokenizer for the source and target language along with generic learning algorithms suffice to produce high quality translations [13]. The technology originally developed to learn transformations from one language or domain to another has applications outside of translation. For instance, statistical MT techniques have shown promising results in the task of generating monolingual paraphrases [15]. Learning label or query transformations is a natural next step.

The labels collected during the game begin in a somewhat different format: for each URL, we have a set of possible labels. We can coerce this data into a reasonable bitext format by simply selecting all pairs: given a URL U and a set of queries $Q(U)$, each pair of queries x_1, x_2 in $Q(U)$ where $x_1 \neq x_2$ is added as to our monolingual constructed bitext. As in the above example of paraphrasing, the source and target languages are identical; we hope to learn transformations that carry approximately the same semantic content using different lexical items.

Given this bitext, we first identify a correspondence between the words in each query pair. This will be used later in extracting single-word or multi-word phrases. As described in [16], the process of finding the most likely alignment between two sequences can be seen as a Hidden Markov Model (HMM). We treat one side of the parallel corpus as the source side, and train a generative model that produces the hidden word alignment and the target side of the parallel corpus using Expectation Maximization (EM) [3]. For each label pair, let s_1 to s_m be the source tokens, t_1 to t_n be the target tokens, and a_1 to a_n be the hidden alignment, where a_i is a value ranging between 0 and m. If $a_i = k$, then target word i corresponds to source work k. Position 0 is a special position indicating the null word. Using the notation s_i^j to represent the sequence $s_i, s_{i+1}, \ldots, s_j$, we can write the probability of a target sequence and word alignment as:

$$P(a_1^n, t_1^n | s_1^m) = \prod_{i=1}^{n} P\left(a_i | a_1^{i-1}, t_1^{i-1}, s_1^m\right) \cdot P\left(t_i | a_1^i, t_1^{i-1}, s_1^m\right)$$
$$\approx \prod_{i=1}^{n} P(a_i | a_{i-1}) \cdot P(t_i | s_{a_i})$$

Basically we assume that there is some jump distribution modeling the position of the source word generating target word i given the position of the source word generating target word i - 1. Also for each source word, we maintain a multinomial distribution over possible target words that are its translation. These distributions are learned from the parallel text using EM as described above. In practice, word alignment quality is improved significantly if we initialize the translation distributions using some simpler model such as IBM's Model 1 [2]. Therefore, we begin by running five iterations of Model 1, and follow up with five iterations of the HMM model described here.

Next we extract so-called *phrase pairs* from the bitext. Note that these phrase pairs are just arbitrary word sequences, not linguistic constituents. We begin by running the word alignment algorithm in both directions: first we select one side of the corpus as the source, train word alignment model, and extract the most likely alignments for each sentence pair; then we flip the assignment and train the models again. The word alignments are combined using a heuristic named Grow-Diag-Final as described in [6]. Let the heuristically-combined word alignment be represented as a relation $\rightarrow$: we say $s \rightarrow t$ if the word s is aligned to the word t. For each query pair $<s_1^n, t_1^n>$, we extract a phrase pair $<s_i^j, t_k^l>$ if (1) there is at least one aligned word pair within that range (i.e., there exists some $x \in [i, j]$ and $y \in [k, l]$ such that $s_x \rightarrow t_y$), and (2) words inside the source phrase only align to words inside the target and vice versa (for all $s_x \rightarrow t_y$ we have $x \in [i, j]$ iff $y \in [k, l]$).

Table 2. Query Alteration Examples

Original Query	Replacement	Category	Probability
jc penny	jc penney	1	0.325
T mobile	T-mobile	1	0.15
acid reflux	acidrefluxconnection.com	2	0.3
zune	zune.com	2	0.3
cbs4	cbs4.com	2	0.3
wayn	where are you now	3	0.3
fai	federal acquisition institute	3	0.3
iht	International Herald Tribune	3	0.3
sls	Society of Laproendoscopic Surgeons	4	0.3
jlo	jennifer lopez	4	0.65
capital city airport	kentucky airport	4	0.3

As an example, consider the pair "*new york department labor*" and "*new york state department of labor*", and assume that all and only the identical words are aligned; thus "*state*" and "*of*" are not aligned. From this pair, we would extract the phrase pairs "*new york* / *new york state*", "*department labor* / *department of labor*", and "*department* / *department of*", amongst many other phrase pairs. However, the phrase pair "*department labor* / *department of*" would not be extracted because some words inside the phrase on one side are aligned to words outside the phrase on the other side.

We extract all phrase pairs consistent with the word alignment from each query pair. To gauge the quality of these phrase pairs, we estimate a probability distribution by aggregating counts across the bitext. Let $c \in [S, T]$ be the count of times we observed the phrase pair S, T from any query pair in the bitext. Then we can define a probability distribution based on the relative frequency of the phrase pairs $(T \mid S) = c(S, T) / \sum_{T'} c(S, T')$.

4.4.2 *Results from Bitext Matching*

For all web pages (URLs) in the Page Hunt pilot data, we extracted the queries that corresponded to winning trials, generated all pairs of queries as bitext data, and applied the algorithm described above to extract all phrase pairs. This results in a large table of alterations (phrase pairs) with weights corresponding to the likelihood of each alteration. Table 2 shows some of the examples. If we take the top scoring alterations, we find that they can be categorized into 4 bins:

1. Spelling or punctuation alterations: For example, we can learn that *JC Penny* can be an alteration for *JC Penney,* or *J C Penny.*
2. Sitename to site alterations: Knowing that *acid reflux* and *acidrefluxconnection.com* are related may be useful.
3. Acronym-Expansion alterations: We can learn that *iht* is the *International Herald Tribune*, *sls* refers to the *Society of Laproendoscopic Surgeons* etc.
4. Conceptual alterations: This is by far the most interesting set. For instance the data we got shows us that *capital city airport* is a valid alteration for *Kentucky airport; jlo* is a valid alteration for *Jennifer Lopez* etc.

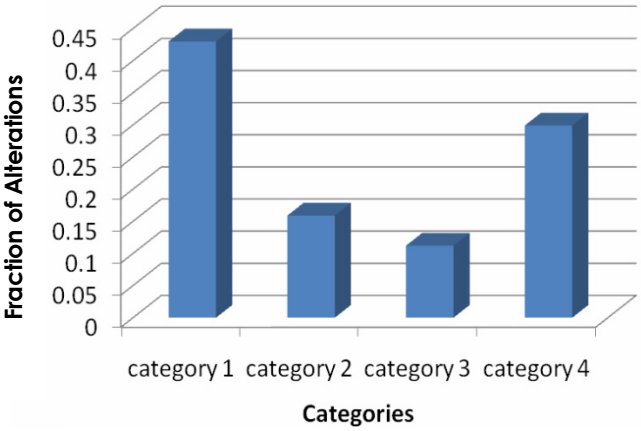

Figure 6. Category Distribution of Query Alterations

Search engines already handle the alterations in the first two categories well. The last two categories, especially the conceptual alterations, look very promising.

In Fig. 6, we also plot the distributions of the query alterations in terms of the categories. From this data, we see that Category 1 and Category 4 alterations are the most frequent alterations we learn from the data.

Our plan is to garner additional data, improve the bitext ranking process, and run our bitext code on the new data to generate alterations. These alterations can then be evaluated by themselves by human judges, and then used in a search engine to see if they cause an improvement in relevance measures.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we present a page-centric approach to improving Web search using human computation games. We provided a detailed description of the Page Hunt game, and briefly described Page Race and Page Match, two variants on the theme. To the best of our knowledge, Page Hunt is the first single-player non-collaborative human computation game.

We showed that the query data we obtain from the Page Hunt game is not very different from queries used on search engines. We discussed the findability metric, which has implications on various aspects of a search engine including indexing, ranking and deduplication. This notion can be used to complement result-relevance metrics and implicit measures of search engine goodness.

We also described a process to extract query alterations from game data, using the idea of bitext matching. There is considerable scope for further work, including evaluations of the utility of query alterations gleaned by this method. We plan to continue work in this area, and to improve our games e.g. by filtering players at the beginning of the game, incorporating player skill levels, dissuading cheating, etc.

In the future, we also plan to conduct an equivalent of eye-tracking analysis from the data we collect.

Eye-tracking analysis can help us with information retrieval tasks related to Web search ranking [25], query expansion [26], etc. However, such experiments not only require expensive equipments like eye-tracking facilities but also require users who participate in the research to be physically present. Moreover, the equipments also require calibration, and are sometimes not accurate enough. Since in our games, players need to scan the whole web page to summarize information, our games potentially record a huge amount of data for analysis.

An example is shown in Table 3 and Fig. 7. Table 3 shows the query logs of URL number 1008 issued by user number 308 in one session, and Fig. 7 shows the content of web page number 1008. We mark the queries on the related location of the web page in time sequence. From this figure, we can observe very clearly the track of this player's attention and also discover how this player extracts useful information from the web page. We can use this information to improve ranking based on page features, and also help users improve their web pages.

Table 3. Logs for Eye-tracking Analysis

User ID	URL ID	Query
308	1008	Apple
308	1008	Apple Store
308	1008	Apple ipod
308	1008	Apple ipod classic

6. ACKNOWLEDGMENTS

In any human computation game, the data comes from the players who gave us data while playing our games. We thank them for their help and the data they provided.

7. REFERENCES

[1] P. N. Bennett, D.M. Chickering, A.Mityagin. Learning Consensus Opinion: Mining Data from a Labeling Game. In WWW '09, pages 121-131, New York, NY, USA, 2009. ACM.

[2] P. Brown, V. Della Pietra, S. Della Pietra, and R. Mercer. The mathematics of statistical machine translation: parameter estimation. Computational Linguistics 19(2): 263-312, 1993.

[3] A. Dempster, N. Laird, and D. Rubin. Likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B, 39(1):1–38, 1977.

[4] D. Edery and E. Mollick. Changing the Game: How Video Games Are Transforming the Future of Business. Pearson Education Inc., 2009.

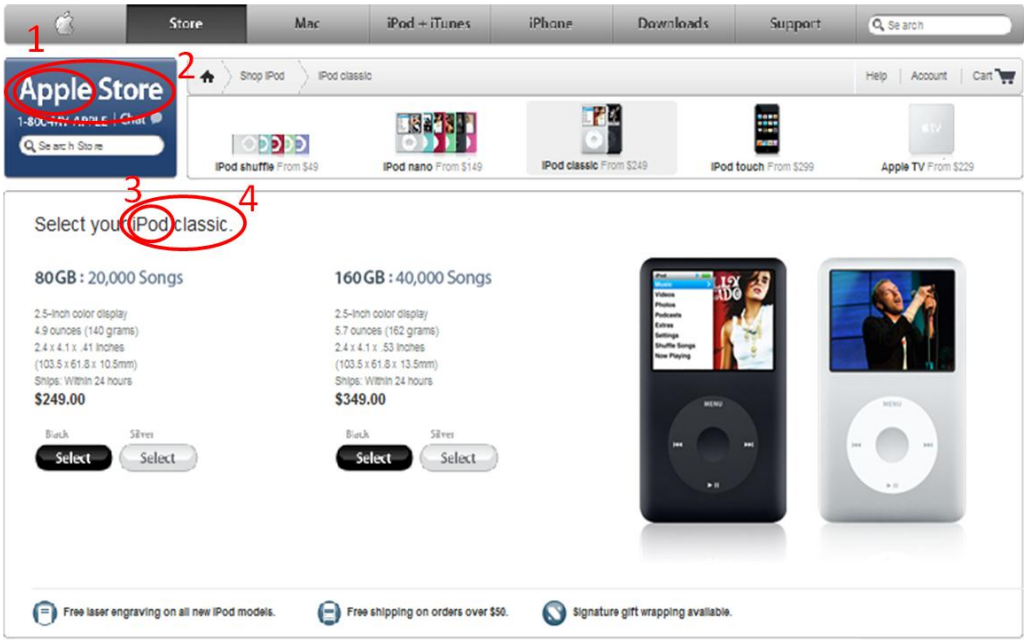

Figure 7. An example of 'eyetracking' in Page Hunt. The red ovals are numbered by the query sequence in the game, in a way tracking the player's visual focus.

[5] N. Ford, T. D. Wilson, A. Foster, D. Ellis, and A. Spink. Information seeking and mediated searching. Part 4. Cognitive styles in information seeking. Journal of the American Society for Information Science and Technology (JASIST), 53(9):728-735, 2002.

[6] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Alberta, Canada, 2003.

[7] E. Law, A. Mityagin, and M. Chickering. Building human-computation games for search. In Submission.

[8] E. Law, A. Mityagin, and M. Chickering. Intentions: A game for classifying query intent. In CHI EA '09, pages 3805-3810, New York, NY, USA, 1980. ACM.

[9] E. Law, L. von Ahn, R. B. Dannenberg, and M. Crawford. Tagatune: A game for music and sound annotation. In Proceedings of the Eighth International Conference on Music Information Retrieval, pages 361-364, Vienna, Austria, 2007. Austrian Computer Society.

[10] T. W. Malone. What makes things fun to learn? Heuristics for designing instructional computer games. In SIGSMALL '80: Proceedings of the 3rd ACM SIGSMALL symposium and the first SIGPC symposium on Small systems, pages 162-169, New York, NY, USA, 1980. ACM.

[11] T. W. Malone. Heuristics for designing enjoyable player interfaces: Lessons from computer games. In Proceedings of the 1982 conference on Human factors in computing systems, pages 63-68, New York, NY, USA, 1982. ACM.

[12] M. McDonald, R. Musson and R. Smith. The Practical Guide to Defect Prevention, Microsoft Press, 2007.

[13] NIST 2008 Machine Translation Evaluation. http://www.nist.gov/speech/tests/mt/2008/doc/mt08_official_results_v0.html

[14] T. Paek, Y.-C. Ju, and Chris Meek. People watcher: A game for eliciting human-transcribed data for automated directory assistance. In INTERSPEECH '07, pages 1322-1325, 2007.

[15] C. Quirk, C. Brockett, and W. B. Dolan. Monolingual Machine Translation for Paraphrase Generation. In EMNLP '04. Barcelona, Spain, 2004.

[16] S. Vogel, H. Ney, and C. Tillmann. HMM-based word alignment in statistical translation. In COLING '96. Copenhagen, Denmark, 1996.

[17] L. von Ahn. Games with a purpose. IEEE Computer, 39(6):92-94, 2006.

[18] L. von Ahn and L. Dabbish. Labeling images with a computer game. In CHI '04, pages 319-326, New York, NY, USA, 2004. ACM.

[19] L. von Ahn and L. Dabbish. Designing games with a purpose. Communications of the ACM, 51(8):58-67, 2008.

[20] L. von Ahn, S. Ginosar, M. Kedia, and M. Blum. Improving image search with phetch. In ICASSP '07, pages 15 - 20, April 2007.

[21] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving accessibility of the web with a computer game. In CHI '06, pages 79-82, New York, NY, USA, 2006. ACM.

[22] L. von Ahn, M. Kedia, and M. Blum. Verbosity: a game for collecting common-sense facts. In CHI '06, pages 75-78, New York, NY, USA, 2006. ACM.

[23] L. von Ahn, R. Liu, and M. Blum. Peekaboom: a game for locating objects in images. In CHI '06, pages 55-64, New York, NY, USA, 2006. ACM.

[24] I. Weber, S. Robertson and M. Vojnovic. Rethinking the ESP Game. Technical Report, Microsoft Research, MSR-TR-2008-132, Sept. 2008.

[25] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In SIGIR '06, pages 19-26, New York, NY, USA, 2006. ACM.

[26] G. Buscher, A. Dengel, and L. van Elst. Query expansion using gaze-based feedback on the subdocument level. In SIGIR '08, pages 387-394, New York, NY, USA, 2008. ACM.

[27] L. Azzopardi, V. Vinay. An Evaluation Measure for Higher Order Information Access Tasks. In CIKM '08, New York, NY, USA, 2008, ACM.

[28] L. Azzopardi, V. Vinay. Accessibility in Information Retrieval. In ECIR '08, pages 482-489, Glasgow, Scotland, 2008.