14

Human Computation Games and Optimization of Their Productivity

Kuan-Ta Chen[1], Chien-Wei Lin[1], Ling-Jyh Chen[1] and Irwin King[2]
Institute of Information Science, Academia Sinica[1] Department of Computer
Science & Engineering, The Chinese University of Hong Kong[2]

Abstract

Web 2.0 technology enables people worldwide to collaborate over the Internet, a phenomenon known as social collaboration. While the incentives for social collaboration are primarily enthusiasm for a particular subject, building a reputation, or gaining a benefit by doing something in exchange for using services or downloading files, the emergence of human computation games has shown that the prospect of having fun can be a strong incentive for participants to actively engage in such collaboration. Among the human computation games, ESP game (ESP stands for Extrasensory Perception) is one of the most popular ones. To play an ESP game, two randomly matched players assign labels that appropriately describe an image provided by the system. It has been shown that the "outcomes" of ESP games have many useful applications, such as image-based CAPTCHA tests and semantic image searches.

In this chapter, we provide an overview of human computation games and present an analytical model for computing the utility of ESP games, i.e., the throughput rate of appropriate labels for given images. The model targets generalized games, where the number of players, the consensus threshold, and the stopping condition are variable. Via extensive simulations, we show that our model can accurately predict the stopping condition that will yield the optimal utility of an ESP game under a specific game setting. A service provider can therefore utilize the model to ensure that the hosted ESP games produce high-quality labels efficiently, given that the number of players willing to invest time and effort in the game is limited.

1. Introduction

The emergence of Web 2.0 has changed the way we solve problems. For example, when we encounter an unknown technical term, it is no longer necessary to consult an expert reference, such as an encyclopedia. Instead, Google and Wikipedia can often provide us with an overview of the term. If we are interested in a book, but we not sure whether it is worth buying, we can go to Amazon to review other readers' comments about it. Similarly, if we are planning a trip, we can visit travelers' help web sites, like Expedia and Orbitz to check which hotels are more comfortable and whether their service is satisfactory. The enhanced ability to share information in the Web 2.0 era can help people make countless decisions in everyday life.

Web 2.0 technology also enables people to collaborate in solving particular real-life problems over the Internet, a phenomenon known as *social collaboration.* One of the most famous examples is Wikipedia, where millions of contributors collaborate in compiling the most complete encyclopedia in the world. Another example can be found in content censoring, which is important for preventing the distribution of inappropriate material on websites that allow users to upload content freely. However, content censoring is labor-intensive for large-scale web services. *Social moderation systems* [1] provide an effective means of reducing the labor required by encouraging users to report inappropriate content. Such users may even become active moderators and develop a reputation in their respective communities. In another example, Internet users can now collaboratively decode the complicated texts of ancient books [8]. Previously, such tasks could only be performed by experts or highly customized OCR (optical character recognition) software. To summarize, with the help of Web 2.0 technology and appropriate designs to motivate people, any group of Internet users, who do not know each other, can combine their "computation power" to solve Al-hard problems. Because of this ability, the process is called *social computation* [2].

Web 2.0 technology enables people to collaborate in solving particular real-life problems over the Internet, a phenomenon known as *social collaboration.* One of the most famous examples is Wikipedia, where millions of contributors collaborate in compiling the most complete encyclopedia in the world. Another example can be found in content censoring, which is important for preventing the distribution of inappropriate material on websites that allow users to upload content freely. To summarize, with the help of Web 2.0 technology and appropriate designs to motivate people, any group of Internet users, who do not know each other, can combine their "computation power" to solve Al-hard problems. Because of this ability, the process is called *social computation* [2].

The key to the success of a social computation system lies in providing sufficient incentives for users to participate in the social collaboration process. The *incentives* provided by most social computation systems are enthusiasm for a particular subject, building a reputation, or even benefiting by doing something in exchange for using services or downloading files. However, it has been shown that the prospect of having *fun can be a strong incentive for people to actively participate in a social computation system.* In [4], Ahn and Dabbish proposed the ESP game, a real-time, web-based, two-player application. To play, in each round, the randomly matched players keep suggesting appropriate labels to describe an image until they achieve a *consensus,* i.e., both players suggest the same label. Since the players are randomly assigned and they cannot communicate, each player must "guess" the labels that his/her opponent will suggest. For this reason, the game is called *ESP (ExtraSensory Perception).* If the players achieve a consensus, the label they agree on is likely to be an appropriate description of the current image. Initially, the ESP game was proposed to acquire users' descriptions for a large set of images, which could then be applied in image-based CAPTCHA tests [3] and semantic image searches. It was later extended to collect users' definitions of the shape of a designated object [7]. The collected results were then used as training data for image understanding and object recognition research.

In [4], Ahn and Dabbish proposed the ESP game, a real-time, web-based, two-player application. To play, in each round, the randomly matched players keep suggesting appropriate labels to describe an image until they achieve a *consensus,* i.e., both players suggest the same label. If the players achieve a consensus, the label they agree on is likely to be an appropriate description of the current image.

In this chapter, our objective is to model the performance of the ESP game and optimize its utility by redefining the criteria for finishing a game. The ESP game proposed in [4] only allows two players to participate. Once they achieve a consensus, the current image is considered solved and the game continues with the next image. In our study, we consider a more generalized ESP game that incorporates the following extensions:

The number of players, n, can be greater than 2.

The consensus threshold, m, can be any positive integer that is not larger than n; that is, a label is considered a consensus decision if it is proposed by m out of n players.

The stopping condition, k, can be any positive integer; that is, an image is considered correctly labeled if k consensuses are reached.

In our framework for generalized ESP games, the game proposed by Ahn and Dabbish [4] corresponds to an instance where $n = 2$, $m = 2$, and $k = 1$. Hereafter, we use "ESP games" or "games" to refer to the proposed generalized version. As some variants of ESP games ask players to label audio clips instead of images, we use the term "puzzle" to denote the target object that players must label by consensus.

In our model, we assume that the number of appropriate labels for each puzzle is limited, and all remaining words are considered inappropriate. For example, to label an image containing a red car beside a river, "car," "river," "red" are considered appropriate or good. Other words are considered inappropriate or bad, even if there is a consensus among the players. For example, players may input typos like "cra," "rive," or "rde" by mistake, or words that are too vague or general, such as "picture," "photo," "sea" and still achieve a consensus occasionally. In such cases, we deem that the current game yields a bad label and the quality of the game's output is decreased.

We model the *utility* of generalized ESP games, i.e., the throughput rate of good labels for the puzzles and its relationship with the game's settings, i.e., the number of players, the consensus threshold, and the stopping condition. We find that a tradeoff exists between the efficiency of the consensus achieved and the quality of matched labels.

Given a fixed number of players and a specific consensus threshold, our model can predict the optimal stopping condition that will ensure the maximal possible utility for an ESP game.

Our contribution in this work is three-fold:

We present a generalized ESP game in which the number of players, the consensus threshold, and the stopping condition are variable.

We propose a probabilistic model that can predict the efficiency, quality, and utility of an ESP game based on the game's settings.

Via extensive simulations, we show that the proposed model can accurately predict the optimal stopping condition, which facilitates the maximal utility of a generalized ESP game. This feature can be used by game service providers to maximize the outcome of games, given that the number of players willing to invest time and effort in the game is limited.

The remainder of this chapter is organized as follows. In Section II, we review related works. We present our probabilistic model for generalized ESP games in Section III, and evaluate its performance via simulations in Section IV. Section V details the optimal stopping conditions predicted by our model. Then, in Section VII, we summarize our conclusions.

2. Related Work

Since Ahn and Dabbish first proposed the concept of the ESP game in [4], a number of social games based on similar ideas have been developed. In the ESP game, players are required to guess the same label for a given image provided by the system. Subsequently, Ahn et al. proposed `Peekaboom` [7], which does not require participants to submit appropriate descriptions for a given image. Instead, players must "circle" a certain object in the image based on a given description. The main difference between an ESP game and `Peekaboom` game is that, in an ESP game, the players guess *what* an image is but they describe *where* an object is in an image in `Peekaboom`.

`Verbosity` [6] collects so-called commonsense decisions of the game's two participants, where one participant acts as the Narrator and the other plays the Guesser. In the game, the Narrator must describe a secret word with a statement comprised of one of the predefined templates and a phrase given by the Narrator. The phrase input by the Narrator cannot include the secret word. The Guesser must guess the secret word based solely on the Narrator's statement. Once the latter makes a successful guess, the system will record the Narrator's statement and use it for further research on knowledge or commonsense reasoning and analysis.

A similar game called `Phetch` [5] is designed to capture users' natural language expressiveness about an image. In this game, a Describer is given a secret image and he/she helps one or more Seekers find the image by describing it with textual statements. The Seekers need to search for the image via an image search engine. Once any of the Seekers successfully identifies the image, the system will record the textual statements given by the Describer and use them as a corpus for research on natural language understanding.

Our work differs significantly from previous studies because we do not propose a new game to test the participants' knowledge. As the ESP game is an effective social computation platform that can "extract" users' knowledge during game play, we believe that it can be "optimized" in terms of "outcomes" through appropriate design. Thus, we focus on how to decide the game settings so that the system can derive more useful information given a fixed amount of resources, i.e., the number of participants. Via analytical modeling, we show that the utility of an ESP game can be maximized by choosing an appropriate stopping condition, i.e., the number of matches needed before a puzzle is considered solved.

3. Modeling of ESP Games

In this section, we describe the proposed probabilistic model for generalized ESP games. First we detail our assumptions and define the variables of the model. We then estimate the number of rounds required to solve a puzzle, as well as the number of good and bad labels suggested by participants before a puzzle is finally solved. Finally, based on our model, we evaluate the productivity of an ESP game by three characteristics, namely, efficiency, quality, and utility.

A. Assumptions

Our model of an ESP game is based on the following assumptions:

1) *Round-based play.* We assume that the game play is round-based rather than continuous. In each round, a player can only make one guess about the current

puzzle, and the system checks whether the players' guesses match at the end of each round.

2) *Independent guess.* For model tractability, we assume that subsequent guesses by a player are independent and identically distributed; that is, a player's current guess is not affected by his/her guesses in previous rounds. Although this assumption somewhat simplifies realistic user behavior, it does not affect the model's accuracy significantly. We discuss this point further in Section VI-A.

Good and bad words. We assume that the number of "good" labels for each puzzle is limited, so all remaining words are considered "bad", i.e., inappropriate. The good words are not known by the game system or the participants a priori. We assume that players will do their best to guess good words in the vocabulary. However, there is a possibility that they will fail to pick the right words; instead, they may make a guess from the bad vocabulary due to a spelling error, a memory error, a misunderstanding, or as a deliberate
ploy.

4) *Uniform guess.* How human beings conceptualize puzzles has yet to be statistically mod-eled. Therefore, we assume that players' guesses are drawn uniformly from both the good and bad vocabulary pools.

Name	Meaning
n	number of players
m	number of guesses required to reach a consensus
k	number of tags required to solve a puzzle
v_{good}	size of the good vocabulary
V_{bad}	size of the bad vocabulary
d	total size of the vocabularies
$prob_{good}$	probability of choosing good words in a round
$prob_{bad}$	probability of choosing bad words in a round

Table 1. Variable definitions

In our model, we assume that n players participate in a game. In addition, the consensus threshold is set to m, and the stopping condition is set to k. For a certain puzzle, the size of the good vocabulary is denoted by v_{good}, while that of the bad vocabulary is denoted by v_{bad}. Thus, the total number of words that players can choose from is $d = v_{good} + v_{bad}$. The probability that a player will guess a word in the good vocabulary is $prob_{good}$; and the probability that a player will guess a bad word is $prob_{bad}$, which is equal to $1 - prob_{good}$. The variables used in the model are summarized in Table I.

B. Time Required to Solve Puzzles

We begin by modeling the number of rounds required to solve a puzzle, i.e., how many rounds it takes to satisfy the specified stopping condition k. The terms "consensus" and "match" are used interchangeably to indicate that a label has been proposed by m players, and denote the label as a *matched label.* In addition, we define a discrete random variable, S, to represent the number of rounds needed to solve a puzzle, and write the probability mass function of S as follows:

$f_S(s)$ = Pr(a puzzle is solved in the s_{th} round) = Pr(no. of matches > k in the s_{th} round),

which is equivalent to

Pr (no. of matches > k in the first s rounds)
$-$ Pr (no. of matches > k in the first $(s-1)$ rounds). We assume the probability that exactly i matches will occur in the first s rounds is $P(i;s)$, and that the i matches will comprise i_{good} matches from good words and i_{bad} matches from
bad words. The number of good matches, i_{good}, must be in the range 0 and $min(i,v_{good})$, and $i_{good} + i_{bad}$ i.
Now we focus on computing the probability of i_{good} matches in the first s rounds. On average, each player in the first s rounds proposes $s_{good} = s \bullet prob_{good}$ good words and $s_{bad} = s \bullet prob_{bad}$ bad words. A match in the first s rounds indicates that at least m players propose the same label in a total of $n \bullet s$ guesses. Moreover, if the matched label is a good word, then the match indicates that at least m players propose that label in a total of $n \bullet s_{good}$ guesses.e can model the probability of one good match occurring in the first s rounds as

$$
\begin{aligned}
&\text{Pr(one good match in the first } s \text{ rounds)}\\
&= P_{good}(1)\\
&= 1-\sum_{q=0}^{m-1}\binom{n \bullet s_{good}}{q}\left(\frac{1}{v_{good}}\right)^{q}\left(1-\frac{1}{v_{good}}\right)^{n \bullet s_{good}-q}
\end{aligned} \tag{1}
$$

Next, we model the probability of i_{good} good matches occurring in the first s rounds. The i_{good} good matches indicate that i_{good} words have been matched, but the remaining $v_{good} - i_{good}$ good words have yet to be matched. Thus, we have a total of C"^{sood} combinations of matched labels. The probabilities of the combinations are equivalent because each word has an equal probability, l/v_{good}, of being selected.herefore, the probability of i_{good} good matches in the first s rounds can be computed by

$$
\begin{aligned}
&P_{good}\left(i_{good}\right)\\
&= C_{i_{good}}^{v_{good}} P_{good}\left(1\right)^{i_{good}}\left[1-P_{good}\left(1\right)\right]^{v_{good}-i_{good}}.
\end{aligned} \tag{2}
$$

Similarly, the probability of i_{bad} bad matches in the first s rounds can be computed by

$$
\begin{aligned}
&P_{bad}\left(i_{bad}\right)\\
&= C_{i_{bad}}^{v_{bad}} P_{bad}\left(1\right)^{i_{bad}}\left[1-P_{bad}\left(1\right)\right]^{v_{bad}-i_{bad}}.
\end{aligned} \tag{3}
$$

Combining Eq. 2 and Eq. 3, we can derive the probability of i matches in the first s rounds as

$$
P(i;s) = \sum_{i_{good}=0}^{\min(i,v_{good})} P_{good}\left(i_{good}\right)P_{bad}\left(i_{bad}\right),
$$

where i_{good} must be in the range 0 and $min(i,v_{good})$ and the sum of i_{good} and i_{bad} must be i. After rewriting the probability mass function of S, the number of rounds needed to solve a puzzle becomes

$$\begin{aligned} f_s(s) &= \sum_{q=1}^{s} f_s(q) \\ &= \left[1-\sum_{i=0}^{k-1} P(i;s)\right]-\left[1-\sum_{i=0}^{k-1} P(i;s-1)\right]. \end{aligned}$$

Finally, we obtain the expected number of rounds needed to solve a puzzle as follows:

$$E(s)=\sum_{s=1} s \bullet f_s(s).$$

C. Number of Matches

In the above subsection, we consider that how many rounds are required for players to achieve consensus on k different labels. Next, we model the composition of the matched labels, i.e., how many good labels and bad labels are matched. First, we derive the expected number of good matches. If we assume that the puzzle is solved in the s_{th} round; then, on average, $n \bullet s_{good}$ guesses will be made by n players, and each of the guesses will be drawn from the v_{good} good words.e treat the question of whether a certain word is a match or not as a Bernoulli event, where "success" indicates that the label is matched and "fail" indicates a non-match. The probability of a good label being matched in the first s rounds is shown in Eq. 1. Consequently, the sum of the Bernoulli random variable of each good word will be a binomial random variable with a success probability equal to Eq. 1. It can be computed as

$$\sum_{v_i \in V_{good}} I(v_i \text{ matched}), \tag{4}$$

where V_{good} denotes the set of good words, and $I(.)$ denotes the indicator function. Let $N_{good}(s)$ be the expected value of Eq. 4, i.e., the expected number of good matches in the first s rounds. The value can be derived by $N_{bad}(s)$, the expected number of bad matches in the first s rounds, can be derived similarly by

$$\begin{aligned} N_{good}(s) &= v_{good} \bullet P_{good}(1) \\ &= v_{good}\left\{1-\left[\sum_{q=0}^{m-1}\binom{n-s_{good}}{q}\right]\left(\frac{1}{v_{good}}\right)^{q} \right. \\ &\quad \left. \times\left(1-\frac{1}{v_{good}}\right)^{n \bullet s_{good}-q}\right]\right\}. \end{aligned}$$

Note that both $N_{good}(-)$ and $N_{bad}(-)$ are functions of S, the number of rounds required to solve a puzzle. In other words, for puzzles that require a different number of rounds to find a solution, the expected number of good matches and bad matches will also be different.Efficiency, Quality, and Utility Here we explain how we evaluate the productivity of an ESP game. We define *the efficiency of an ESP game as the rate that labels are matched for the given images.Ifthe* number of participants remains the same, higher efficiency indicates that

the system is more "productive" given the same amount of resources. Thus, we consider game settings that lead to higher efficiency as more desirable.n addition, we define *the quality of an ESP game as the proportion of good labels among all the matched labels.* Higher quality indicates that the matched labels are more likely to be appropriate descriptions of the target puzzle. Thus, we naturally seek game settings that yield high-quality matched labels.However, there is often a trade-off between efficiency and quality in a real system because configurations that yield higher efficiency often lead to lower quality; conversely, settings that yield higher quality may impact on the level of efficiency. For this reason, we define *the utility of an ESP game as the product of the game's efficiency and quality.* This definition enables us to explain utility as *the throughput rate ofgood labels produced by an ESP game.*
Based on the probabilistic model presented in this section, we can write the formula of the efficiency, quality, and utility of an ESP game as follows:

$$\begin{aligned} Efficiency &= \frac{E\left(N_{good}(s) + N_{bad(S)}\right)}{E(s)}; \\ Quality &= \frac{E\left(N_{good}(s)\right)}{E\left(N_{good}(s) + N_{bad}(s)\right)}; \\ Utility &= \frac{E\left(N_{good}(s)\right)}{E(s)}. \end{aligned} \tag{5}$$

Name	Default value	Name	Default value
n	**2**	d	**1000**
m	**2**	$prob_{good}$	**0.8**
v_{good}	**20**	$prob_{bad}$	**1 - $prob_{good}$**
V_{bad}	d v_{good}	T	**10000**

Table 2. Default values of variables

4. Model Validation

In this section, we describe the simulations used to validate our model. After explaining the simulation setup, we compare the utility computed by our model with that derived in the simulations. The effects of various game parameters on the game's utility are also considered.

A. Simulation Setup

We designed our simulator based on the rules of ESP games. In each round, there are n players, each of which randomly selects a good word with probability $prob_{good}$, and a bad word with probability $prob_{bad}$. At the end of each round, the simulator checks the number of matches to determine whether the current puzzle has been solved. If m matches are found, all the players' guesses are erased to simulate that the participants are trying to solve a new puzzle; otherwise, the simulator just advances to the next round. The simulator assumes the number of puzzles is infinite, and that there are always n players ready to participate in a game. The simulations end after running for T rounds, no matter how many puzzles have been solved. We then compute the average efficiency, quality, and utility of the matches based on the time taken to solve each puzzle and the number of good and bad matches recorded during the simulations.

To investigate the accuracy of our model under different settings, we change the parameters and observe whether the simulated quantity of good and bad matches is identical to or close to that computed by our analytical model. Specifically, we change the four major variables, i.e., the number of players, n; the consensus threshold, m; the size of the good vocabulary, v_{good}; and the probability that the players will guess a good word, $prob_{good}$. When evaluating the effect of one variable, the other three are set to their default values, as shown in Table II. Moreover, when we adjust the consensus threshold, we set the number of players at 20, as the consensus threshold must be no greater than the number of players.

B.Validation by Utility Curves

Although we have defined three key characteristics of an ESP game, namely, the efficiency, quality, and utility, we only validate the accuracy of our model by a game's utility. This is because the magnitude of the utility depends on the efficiency and the quality; thus, the utility is unlikely to be correct if the values of the other two characteristics are incorrect. Since our objective is to optimize the utility of ESP games by changing the game settings, the model's accuracy in predicting a game's utility should be examined more carefully.

In the following, we investigate how the utility of an ESP game changes under different stopping conditions, k. As shown in Fig. 1, the utility reaches its maximum when $n = 2$ and $k = 10$. As the number of participants increases, the shapes of the utility curves change slightly, and the optimal stopping condition shifts slightly to the lower k values. The concave shape of the utility curve indicates that, as k increases, there should be a tradeoff between the efficiency and quality of ESP games such that the utility curve is not monotonic. To demonstrate the tradeoff between efficiency and quality, we plot the values of all three characteristics in Fig. 2. Clearly, the game's efficiency increases as k increases, while its utility decreases. The utility reaches the highest point when k is around 15.

We now consider the effects of the other parameters on the utility curves of ESP games, and check the correspondence between the results derived by our model and those of the simulations. The effects of the consensus threshold, the size of the good vocabulary, and the probability that players will guess a good word are investigated. However, because of space limitations, Fig. 3 only shows the effect of the consensus threshold. For all the parameters, the utility curves computed by our model are very close to those derived by the simulations. We observe that the consensus threshold and the size of the good vocabulary have a strong effect on the optimal stopping condition, while the number of participants and the probability of choosing good words have relatively little effect.

C.Effect of Game Settings

We now examine the effect of various game settings on the game's utility. The relationships between the utility and different game parameters are shown in Fig. 4. Figure 4(a) shows that if more players participate in a game simultaneously, the matching rate of good words increases faster than linearly, as the number of guess-pairs grows quadratically. In contrast, if the consensus threshold is raised, as shown in Fig. 4(b), the game's utility declines exponentially, but the quality of the matched results increases. The size of the good vocabulary also has a substantial impact on the game's utility. Figure 4(c) shows that the utility gradually decreases as the size of the good vocabulary increases because of the lower probability that two participants will guess the same good word. Finally, as expected, the game's utility increases linearly as the probability of guessing good words rises. Note that, in all the graphs, the utility scores computed via simulations and by our model match closely, which demonstrates the accuracy of our analytical model.

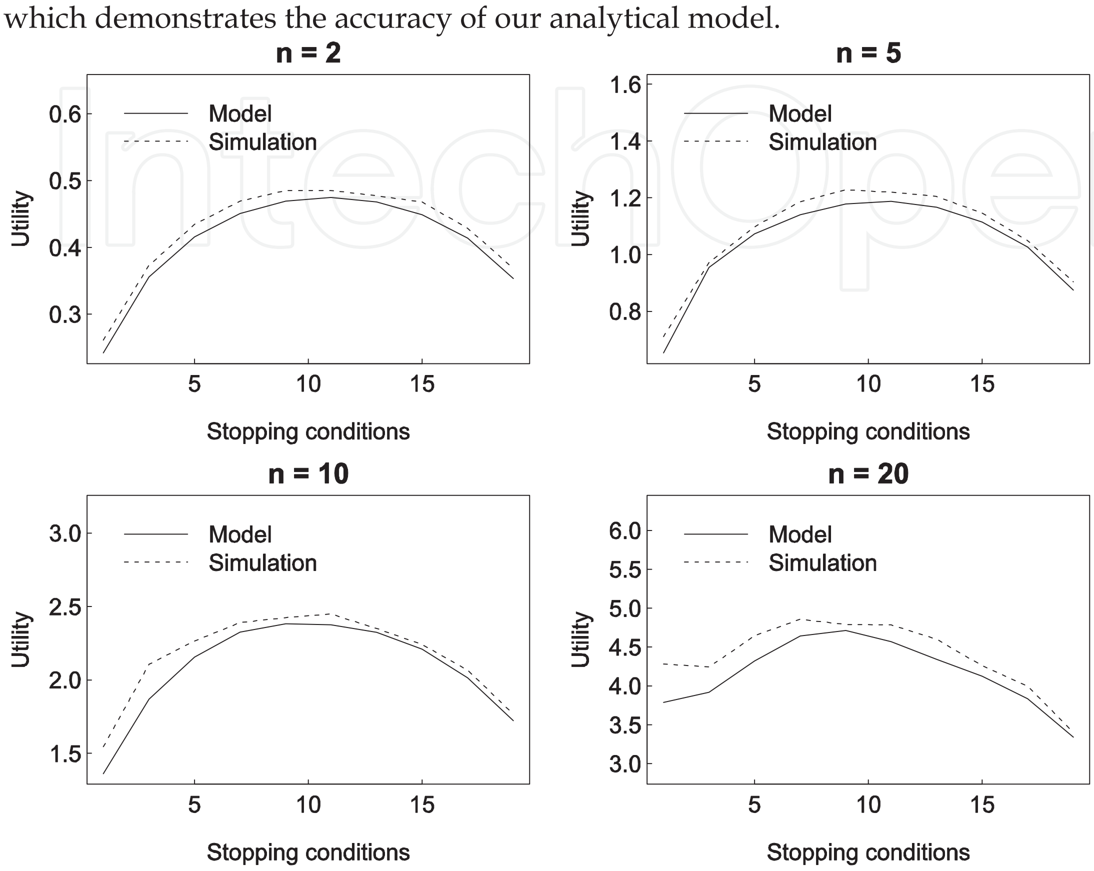


Fig. 1. The relationships between utility and stopping conditions under different n.

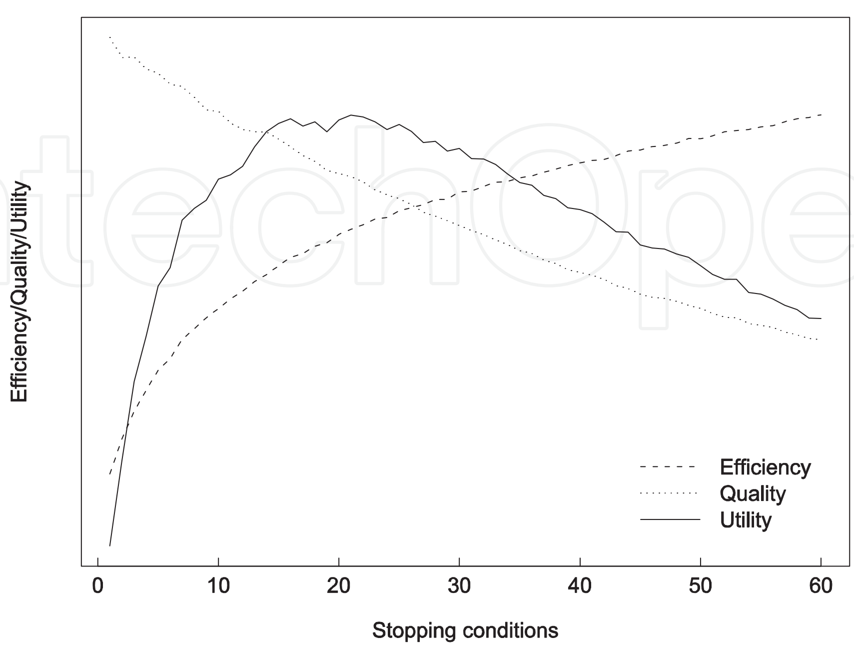


Fig. 2. The relationships between efficiency, quality, and utility in an ESP game.

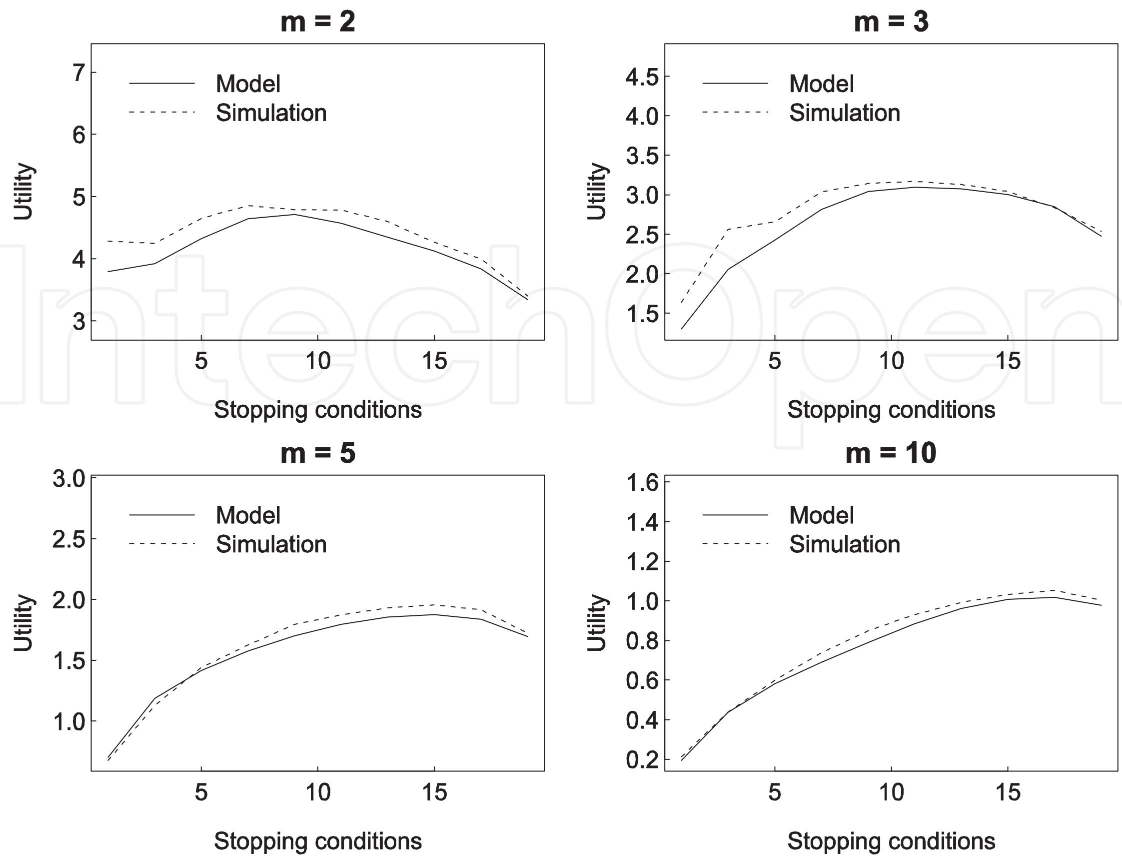


Fig. 3. The relationships between utility and stopping conditions under different m.

5. Optimal Stopping Conditions

In this section, we focus on how to set the stopping condition to maximize an ESP game's utility. We explain the derivation of the optimal stopping conditions, and discuss how they change under different configurations. In addition, we examine how our optimization method improves the game's utility.

A. Computation

The utility equation of our model (Eq. 5) is a discontinuous function, so we cannot obtain its optimal point by differentiating the function with respect to the stopping conditions. Therefore, we derive it in a numerical way. From Section IV-B, we know that the utility function that takes the stopping condition, k, as the only parameter is a unimodal function. In addition, the domain

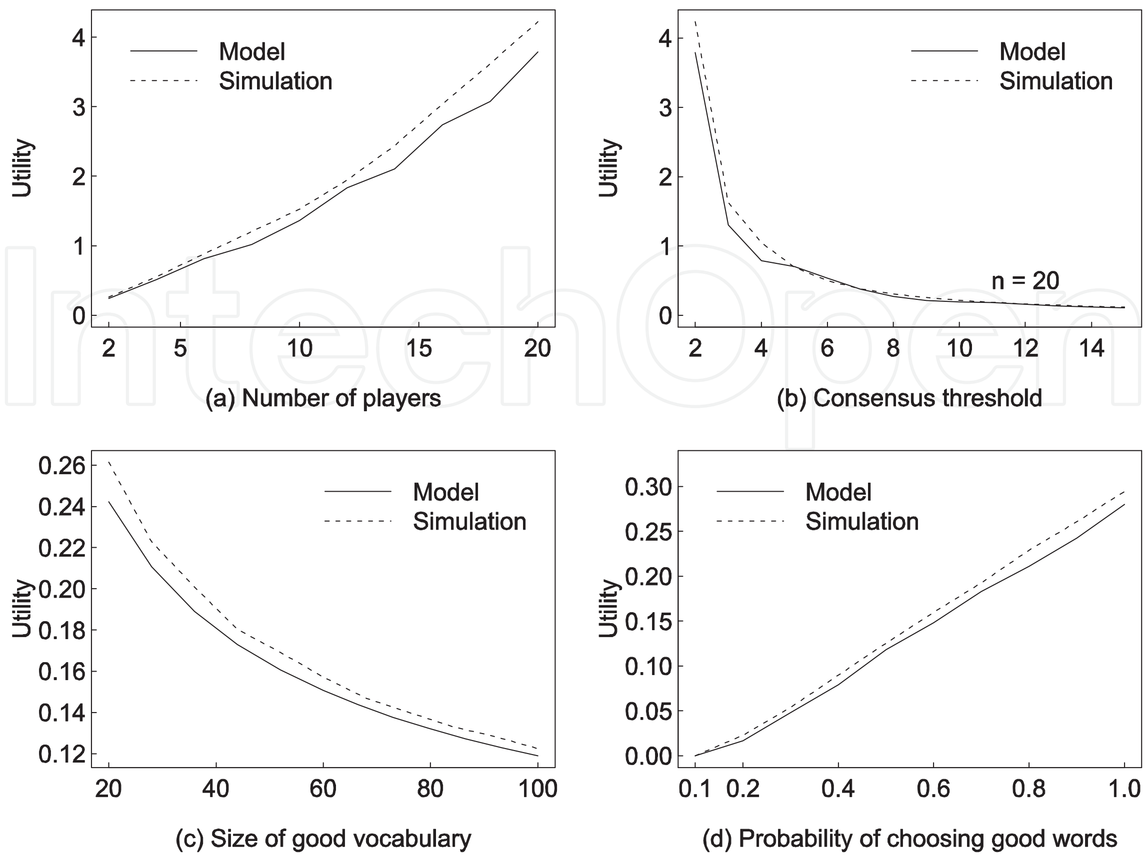


Fig. 4. The effect of other parameters on utility.

of k is a positive integer, which is usually small (less than 100 in most of our scenarios). Thus, we use an exhaustive search to find the maximum utility within a reasonable range, say, from 1 to 200. In our implementation, this exhaustive search process takes only a few seconds on a commodity PC.

B. Effect of Parameters

Here, we consider the effect of different parameters on the optimal stopping conditions. Interestingly, the number of participants does not affect the optimal stopping conditions, as shown in Fig. 5(a). This is reasonable because the probability of good matches and bad matches remains the same regardless of the number of players, which only affects the rate of label matching. The consensus threshold, on the other hand, affects the optimal stopping conditions significantly when it increases, as shown in Fig. 5(b).

This behavior can be explained by the occurrence probability of good matches relative to that of bad matches. Raising the consensus threshold makes label matching more difficult; however, the advantage is that matching bad labels will become relatively more difficult than matching good labels. Therefore, when the consensus threshold increases, the matching rate of good labels will grow faster than that of bad labels; consequently, the optimal stopping condition is deferred to allow more good words to be matched before finishing the puzzle.

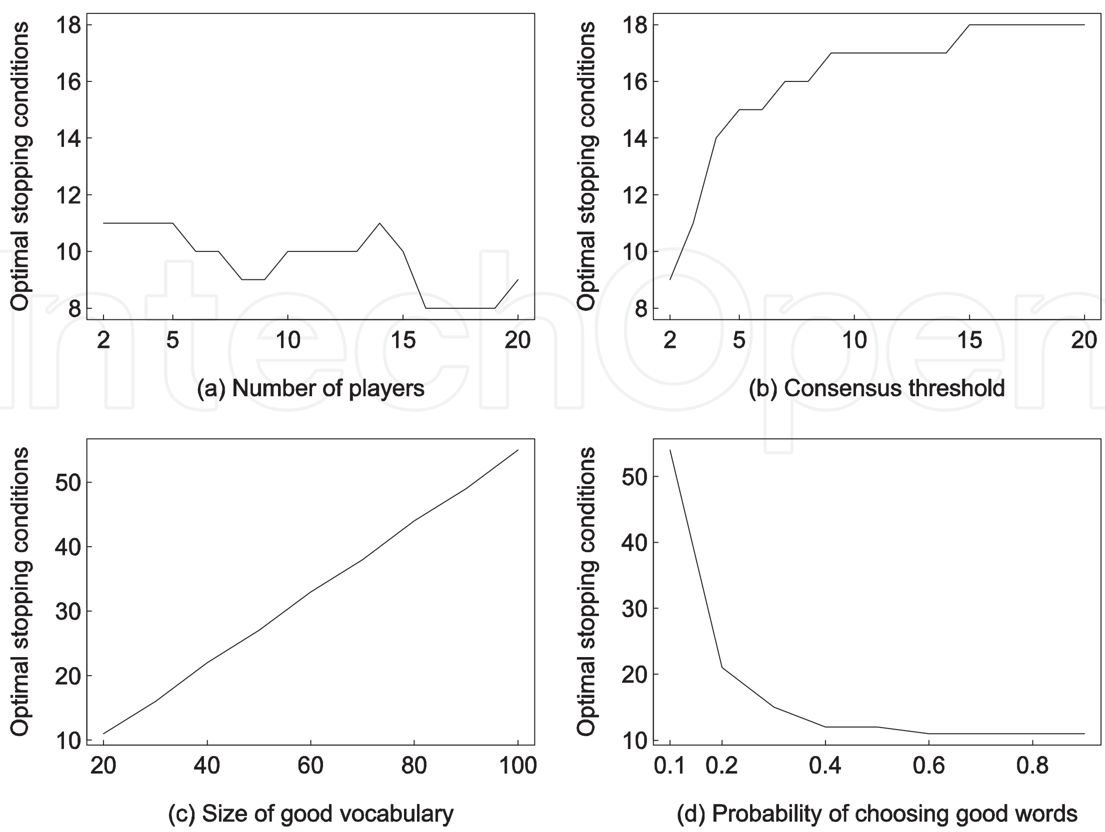


Fig. 5. The effect of the parameters on the optimal stopping conditions

The size of the good vocabulary and the probability of choosing good words have similar impacts on the optimal stopping conditions. Both increasing the number of good words and reducing the probability of choosing good words increase the optimal stopping conditions because they make matching good labels more difficult. Thus, a relatively late stopping condition is required in order to increase the proportion of good matches.

C. Benefit of Optimization

To demonstrate how optimization improves the game's achieved utility, we examine the gain derived by adopting the optimal stopping condition suggested by our model. We define the *utility gain* as the ratio of the utility of an optimized game to that of a simple ESP game, i.e., with the stopping condition set to 1.

The relationships between the utility gain and various game parameters are shown in Fig. 6. We observe that, the optimization achieved by adopting the optimal stopping condition generally provides a utility boost that is 2 or more times higher than that of the simple ESP game. Even if we consider a more conservative scenario, where only two participants play the game and the consensus threshold is set to 2, the utility gain will be around 2, assuming the number of good words is 20 and the probability of choosing good words is 0.8. Moreover, the utility gain increases rapidly as either the consensus threshold or the size of the good vocabulary increases. The utility gain is only significantly lower than 2 when the number of participants is much higher than 2. However, we can still achieve a utility gain of around 1.3, even the number of players is as high as 20. These findings demonstrate that the utility optimization provided by our analytical model can generally provide twice as much utility as a non-optimized game, which stops immediately after a label has been matched.

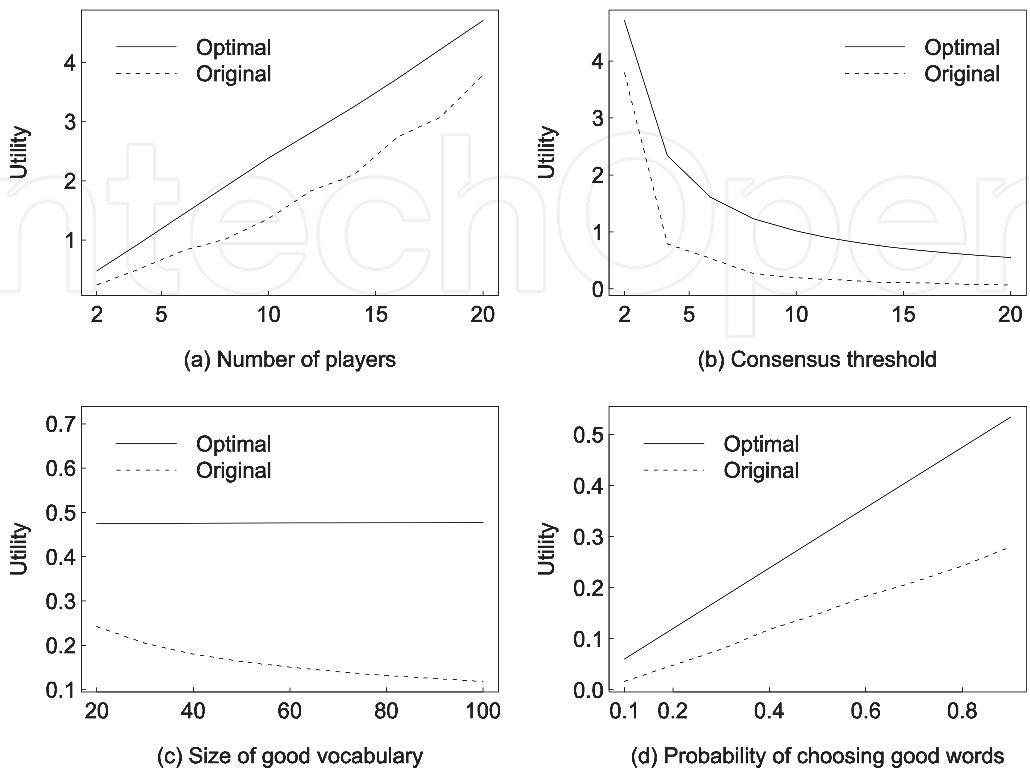


Fig. 6. The effect of the parameters on the improvement in utility

6. Discussion

In this section, we discuss the effects of the assumptions used by our analytical model and some issues that may occur when applying our optimization technique in real-life ESP games.

A. Model Assumptions

One major assumption of our analysis is that the guesses made by each player are independent of each other. in practice, players remember what labels they have already used and avoid submitting duplicate guesses. However, considering the "memory" effect would make the analytical modeling too complicated to manage. Thus, we adopt the independent guess assumption and examine its impact on the model's accuracy by simulations.

To demonstrate that our model provides a reasonable solution for utility optimization, we show the optimal utility achieved by different models and simulations respectively in Fig. 7. Because we do not actually construct an ideal model that takes the memory effect into consideration, we compute its output by simulations. On the graph, the three curves in the figure represent the optimal utility achieved by the ideal model, by our model with the independent guess assumption, and by simple games in which k is set to 1. The results show that both models yield much higher utility than the simple games. Even though our model does not provide as high utility as the ideal model, the games that adopt the stopping condition suggested by our model still achieve near optimal utility. In view of the complexity of modeling with the memory effect, we consider that our independent guess assumption is a reasonable tradeoff between the model's computational complexity and the degree of optimization we are pursuing.

Another assumption of our model is that players uniformly guess words in the vocabulary pool. In practice, players may guess according to some preferences. For example, they may prefer to guess more common, shorter words first, or guess more specific words first, because they think a particular strategy would lead to consensus more quickly. Players' strategies in prioritizing their label choices may significantly impact the outcome of an ESP game. In addition, the situation becomes more complex when players with different strategies are assigned to the same game. Thus, we leave the modeling of players' strategies for choosing words to a future work.

B. Choice of Parameters

To put our model to real use, we must first address the problem of how to choose the model's parameters, especially the size of the good vocabulary and the probability that players will guess a good word. We believe that these parameters could be measured *empirically* from reallife observations. Specifically, one can take the average number of labels on which there has been a consensus in a large number of games as the size of the good vocabulary. Accordingly, one can compute the probability that players will guess a good word by the ratio of guesses that fall into the set of the good vocabulary. While the parameters may be different due to the types of puzzles and the composition of the participants, an empirical choice of parameters like this would be the most appropriate way to achieve accurate modeling results and thereby optimize the utility of games.

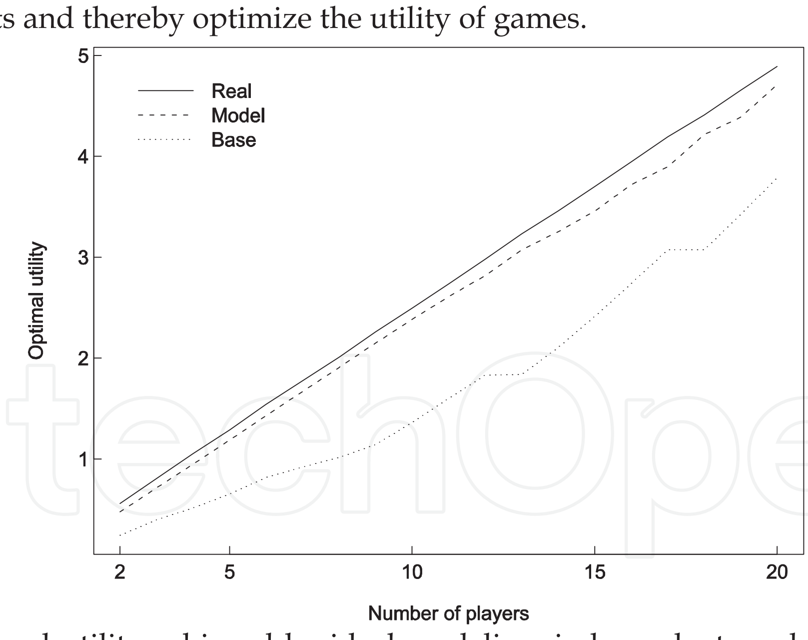


Fig. 7. The optimal utility achieved by ideal modeling, independent modeling, and simple games without optimization.

7. Conclusion

We have proposed a generalized ESP game in which the number of players, the consensus threshold, and the stopping condition are variable. In addition, we have presented an analytical model that computes the efficiency, quality, and utility of an ESP game given the game's settings. Via extensive simulations, we show that by applying the optimal stopping condition predicted by our model, the game's utility will be usually be at least 2 times higher than that of a non-optimized game. This feature can be leveraged by game service providers to improve the utilization of finite player efforts in order to maximize both the efficiency and quality of the matched labels.

8. References

[1] C. Lampe and P. Resnick, "Slash (dot) and burn: distributed moderation in a large online conversation space," *Proceedings of the 2004 conference on Human factors in computing systems,* pp. 543-550, 2004.

[2] D. Stork *et al.,* "The Open Mind Initiative," *IEEE Expert Systems and Their Applications,,* vol. 14, no. 3, pp. 19-20, 1999.

[3] L. von Ahn, M. Blum, N. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Proceedings of Eurocrypt.* Springer, 2003.

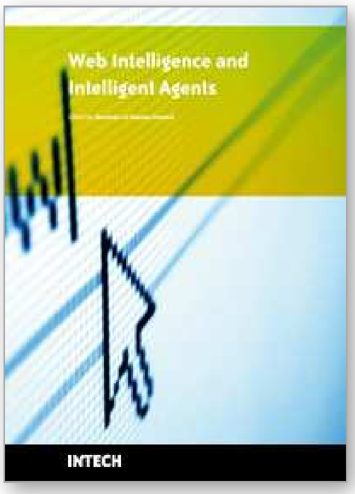
[4] L. von Ahn and L. Dabbish, "Labeling images with a computer game," *Proceedings of the SIGCHI conference on Human factors in computing systems,* pp. 319-326, 2004.

[5] L. von Ahn, S. Ginosar, M. Kedia, and M. Blum, "Improving Image Search with PHETCH," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on,* vol. 4, 2007.

[6] L. von Ahn, M. Kedia, and M. Blum, "Verbosity: a game for collecting common-sense facts," *Proceedings of the SIGCHI conference on Human Factors in computing systems,* pp. 75-78,2006.

[7] L. von Ahn, R. Liu, and M. Blum, "Peekaboom: a game for locating objects in images," *Proceedings ofthe SIGCHI conference on Human Factors in computing systems,* pp. 55-64,2006.

[8] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "reCAPTCHA: Human-based character recognition via web security measures," *Science,* vol. 321, no. 5895, p. 1465,2008.


Web Intelligence and
Intelligent Agents
INTECH