CIME: some protocols for Computer In the Middle Education

Jérémy Barbay

David R. Cheriton School of Computer Science
University of Waterloo, Canada.

Technical Report CS-2007-19

Abstract. We introduce *Computer In the Middle Education* (CIME), a family of software providing access to some educational data in exchange of further addition to this data. We present several prototypes: `VocaCIME`, focused on gathering and teaching foreign vocabulary associated to images; and `CIMEQuiz`, focused on multi-choice questions on simple mathematical concepts. In particular, we describe techniques for the *quality control* of the data submitted by the user, and techniques to insure the growth of the database when it is used. While this work is partially inspired by the human computation games `ESP` [2], `Peekaboom` [5], `Phetch` [3] and `Verbosity` [4] from Luis von Ahn and his collaborators, the approach illustrates new directions of research, such as *bootstrapping databases*, where a small initial database is grown into a larger one by its mere use, and *computer in the middle* techniques, where the computer provides a communication interface between users (rather than a game), but still recovers useful data from it. This technical report describes prototypes under development or yet to be developed.

Keywords: ESP, Peekaboom, Phetch, Verbosity, Human Computation, bootstrapping, Computer in the Middle, Language Educational Software, Database of Solved Questions, Quizzes.

1 Introduction

Language educational software such as "`Rosetta Stone`" or "`Before You Know It`" are based on some very simple software, and on expensive databases of images and sounds indexed in foreign languages. Note that although there already exists extensive image indexes in English [2], and dictionaries translating words between English and foreign languages, combining those to obtain image indexes in foreign languages yields poor results: most languages are more specific than English on one concept or another, and this kind of translation of the index loses most of the nuances, which are important when learning a foreign language. We describe with `VocaCIME` how such a database can be bootstrapped from a small initial set of data into a database which grows in size and quality as more people use it, through a combination of games of recognition. Note that those techniques not only applies to the labeling of images in other languages than English, but also potentially applies to any other media such as sound or even video. We describe with `CIMEQuiz` how more complex data (e.g. more complex foreign sentences, mathematical concepts) can be similarly boot-strapped, through the use of template questions with multiple answers.

Luis von Ahn and his collaborators [2, 5] studied through various projects how to use human computation in order to solve problems which are hard for computers but easy for humans, through online games. They use a variety of techniques to check the quality of the data submitted, and the seriousness of the user, in particular to avoid spam. The techniques of VocaCIME extend the techniques illustrated by Peekaboom in the sense that users “pay” their access to the data, either by creating additional data, or by checking the quality of data submitted by others. While the contribution of the user still has to be short and concise, this approach is more flexible than games as it allows slightly more “boring” tasks to be assigned to users. The techniques of CIMEQuiz extend the techniques illustrated by Verbosity [4], as the more general type of questions allows to combine the answers of different questions; and as the combination of three roles in two games (instead of two roles in one game for Verbosity) allows to consider questions with an unbounded number of answers.

2 Prototypes

We describe in this section some software using the concepts of “computer in the middle” and “bootstrapping”, and various techniques to insure the quality of the data added to the database. We will review the various techniques used in Section 3.

2.1 VocaCIME

We present VocaCIME, an example of software providing the user with the access to some database of images indexed in many different language, in exchange of additions to the database or of some quality control of the data contained in the database.

At the beginning of each session, the user identifies in a list a language in which he considers himself to be an expert, and the language he desires to learn. Further on the user will be tested on its expertise for the first language, and will pay for his access to the data related to the second language by contributing data and/or validation in the first language.

The session is then iterating cyclically through two phases: the *expert phase* during which the user contribute to the database, and the *learner phase* during which the user accesses the databases in a restricted way. Each phase can be realized in several variants, which are presented below. We do not present the variants by type of phase but rather by order of complexity, for the sake of readability.

Expert Phase of input: During this expert phase, the user labels images in his language of expertise. The goal is to input the basic data which will be used to teach learners in this language.

As a single image can represent several terms, for each term α (e.g. dog) the expert is presented with four images which are all associated with α, but so that the intersection of the terms associated to each labels is reduced to $\{\alpha\}$ (see Figure 1). Such exhaustive association of terms with images is produced by tools like ESP[2][1].

The expert labels several terms during a single phase, including a few randomly chosen which are already validated. The information entered by the expert is accepted and added to the database only if it agrees with the data already validated. This data will be later validated only if a majority of experts entered the same term. The expert is allowed to enter the following learner phase only if the information entered has been accepted.

[1] Alternatively, the information produced by a tool such as Peekaboom [5] can be used to produce a single image where the term to be entered is the only one present.

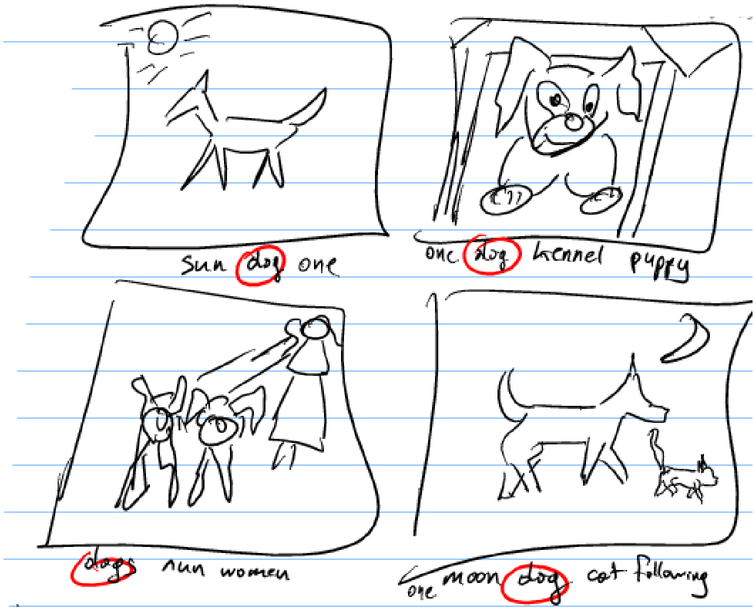


Fig. 1. Expert Phase, input

Learner Phase of one term mapping: The goal of this phase is to check the memorization ability of the learner.

At the beginning of the phase the learner selects from a list a concept that he wishes to learn (e.g. “animals”, “numbers”). Each concept of the list corresponds to a prepared list of terms relevant to the concept and for which there are relevant images indexed. The phase iterates randomly in this list in rounds, each round teaching and testing the knowledge of the learner on one specific term.

During each round of the learner phase the learner is presented with one term in the learned language and four images, one image illustrating the term given and not the three other images (see Figure 2). The goal of the learner is to select the image corresponding to the term, by *recognizing* the term in the learned language.

When the learner selects directly the correct image, the term is validated and not reused later. When the learner selects a wrong image, the term corresponding to the selected image is shortly displayed, and the term is tagged to be revisited later in the session, potentially with different pictures.

Given a list of concepts already validated by the learner during this session, the software generates lessons such that the answer of a round can be *deduced* using the knowledge or previous results. For instance, knowing that the learner acquired the concept of numbers in the past and requested a lesson on animals, the term corresponding to “five ducks” in the learned language (e.g. “cinq chiens” in French) can be asked in conjunction with images representing “one dog”, “two cats”, “five ducks”, and “10 birds”. The learner can then use previous knowledge (“cinq”→five) to recognize the right image and acquire new knowledge (“chiens”→dog) at the same time, making for a learning experience more based on success.

Learner Phase of one image mapping: The goal of this phase is to check the reading abilities of the learner.

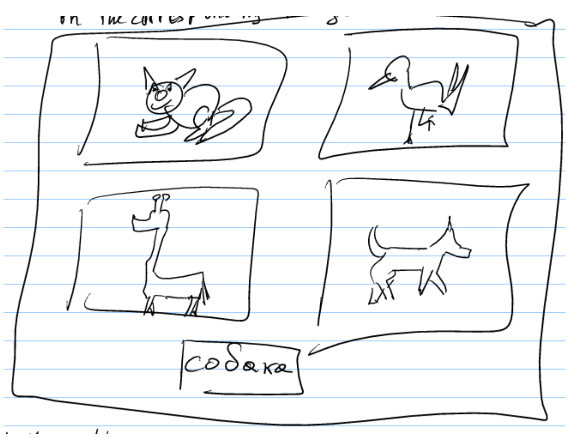


Fig. 2. Learner Phase, one term mapping

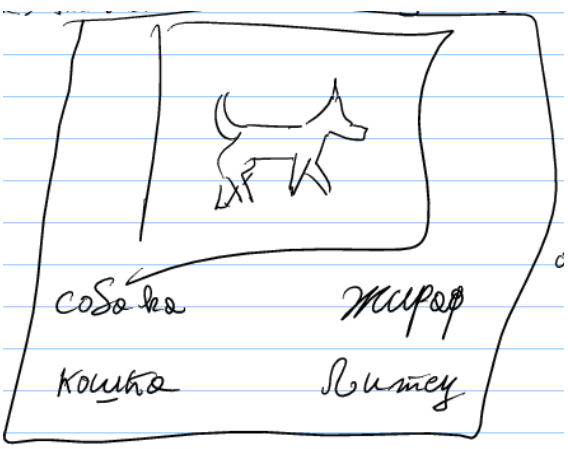


Fig. 3. Learner phase, one image mapping

In a symmetric way to the one term mapping learner phase, during each round the learner is presented with one image and four terms in the learned language, one term illustrating the image given and not the three others (see Figure 3). The goal for the learner here is to understand the terms enough to determine between them the one which corresponds to the image.

When the learner selects directly the correct term, this term is validated and not reused later. The management of errors is slightly different from the one term mapping learner phase. When the learner selects the wrong term, both the correct term and the term wrongly selected are tagged to be revisited later in the session. As in the one term mapping learner phase, the image corresponding to the term wrongly selected is displayed, so that the learner can learn from his mistakes and can converge to the right answer.

Learner phase of input: The goal of this phase is to test the writing of the learner. Similarly to the input expert phase, the learner is presented with four images having one term in common, and asked to type the corresponding term. If the term mismatches the term associated to the four images, the learner is presented with the solution and the "error" is recorded in the database for later use in a expert phase of validation (see Section 2.1).

Expert phase of validation: The goal of this phase is to require the expert to answer what appeared as mistakes during a learner phase. Some might be mistakes on which the expert can comment for future learners doing the same mistake; and some might not be mistakes, but admissible variants, which are entered in the database so that further learners false a more clever electronic teacher.

During each round, the expert is given four images and a term α which potentially mismatches the term β associated to the four images, such as a term typed in a learner phase of input. The expert must then decide if α is still a valid answer, and comment on his decision.

The expert comments on several mistakes during a single phase, among which the answer of a few is already known and used to check the validity of his validation. The validity of his comments is checked by another expert phase (see Section 2.1).

Expert phase of double validation: The goal of this phase is to validate the comments entered in an expert phase of validation, which constitute another form of free data submitted for addition to the database. This case is different of the data submitted in an expert phase of input, where the data consists of a single term which can be compared with other submissions and put to a majority vote to validate it.

The expert is presented with a mistake and several comments entered in an expert phase of validation. It must approve the comments, and as before the validity of some comments is already known to check his commitment, and his validation will be put to a majority vote before it get validated.

Future Work

1. Obviously, it would be possible to extend the technique to other medias, such as short sounds or videos, asking the expert to identify terms in those videos. Adding sound would permit to teach the pronunciation of words (although testing it might require much more work). Adding video would permit to manipulate better some concepts more complex than terms: some concepts can be expressed awkwardly through pictures (e.g. "the boy jumps from the table", "the turtle won the course")
2. With some higher risk for abuse (and hence requiring some more advanced quality control techniques, in particular such as the identification of the users), it would be possible to require the expert to input some sound, video or pictures. This will require much more advanced techniques of quality control, which we discuss in Section 3.
3. Another extension requiring more advanced techniques is to replace the hard written "lessons" describing in English which terms should be taught together by generated ones, such as ontologies. Such lessons could be generated using machine learning techniques, or alternatively entered by one expert, and validated by other experts through a majority vote. This would also require more advanced techniques of quality control.

2.2 CIMEQuiz

We present `CIMEQuiz`, a piece of software which permits students to test their understanding of the basic material of a course, while acquiring from them more testing material.

The techniques used are similar to the templates illustrated by `verbosity` and more generally to the asymmetric verification games illustrated by both `Peekaboom` [5] and `Verbosity`. Our techniques are different in that they apply to questions with an unlimited set of answers (whereas asymmetric verification games require them to be bounded), through the combination of two complementary games on the same data.

Both games are based on a few templates of multi-choice question, parameterized by a *domain* and a *property* (see Figure 4 for some examples): "Among the following `XXXXXXX`, which ones are `YYYYYYY`?" and its complement "Among the following `XXXXXXX`, which ones are **not** `YYYYYYY`?", "Give 10 examples and counter-examples of `YYYYYYY` among `XXXXXXX`.", "Which domain and property are identified by those examples and counter-examples?".

The games can be played by two users connected at the same time and randomly paired (as for ESP or Peekaboom), or can be played offline, the computer taking the role of any player. Each player can assume one of three roles, A, B or C. The player in the role A inputs some positive and negative examples of objects verifying the property; the player in role B tries to guess the property

Among the following `integers`, which ones are `prime numbers`?

(y) 1	(y) 2
(y) 3	(n) 4
(y) 5	(n) 6
(y) 7	(n) 8
(y) 255	(n) 257

Among the following `countries`, which ones are `in the European Union`?

(y) France	(y) England
(n) Switzerland	(y) Italy
(y) Germany	(y) Romania
(n) Croatia	(n) Turkey
(y) Spain	(y) Luxembourg

Among the following `properties on asymptotics`, which ones are `correct`?

(y) $2^n \in O(3^n)$	(n) $2^n \in \Omega(3^n)$
(n) $2^n \in \Theta(3^n)$	(y) $n^2 < n^5$
(n) $n^2 \in \Omega(n^5)$	(y) $n^2 \in O(n^5)$
(y) $\lg n \in O(n)$	(y) $n \in \Omega(\lg n)$
(n) $\lg n \in \Omega(n)$	(y) $n \lg n \in \Omega(n)$

Among the following `integer functions`, which ones are `in` $O(n^5)$?

(n) 2^n	(n) 3^n
(y) n^3	(y) n^2
(n) $(n^5)^2$	(y) $n \lg n$
(y) $5n^5$	(y) $(\lg n)^5$
(y) $\lg(32^n)$	(n) $\lg(33^n)$

Fig. 4. Some examples of questions, and of their possible answers in the database. Not shown here, the validity of each answer and question can be currently *validated* or *tested*.

from a set of positive and negative examples; and the player in role C just has to decide among a set of objects which ones verify the property. The games are asymmetric, in that each player has a different role: the first game assigns the roles A and B, while the second game assigns the roles A and C.

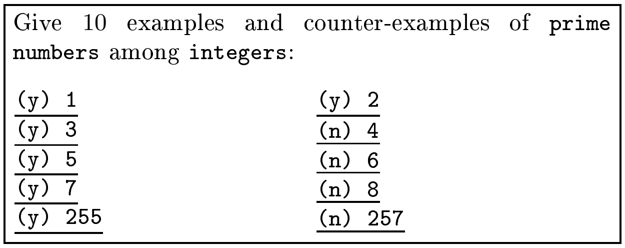
As the probability of two people wanting to play on the same topic at the same time can be small, it is important to support versions of the game for a single player, where the computer assumes a role. During each session the player is forced to cycle randomly among the three roles, and the player assigned to role A does not know which game he is playing (hence he does not know which role is assigned to the other player), nor if the other role is assumed by the computer or a human. We describe those roles below, how they combine in two games which produce and validated new data for the database, and how each role can be assumed by a player or the computer.

Role A: input The player with role A must input 10 examples and counter-examples of objects verifying a given property in a given domain (see Figure 5). This is the primary source of input in the database, and the quality of the player input cannot be checked using the techniques of asymmetrical verification illustrated by `Peekaboom` [5] and `Verbosity` [4], because the number of possible answers is too large, or even potentially infinite for some domains.

If the other player is in role B, both players are collaborating and both their score is increased if B discovers the concepts in a few trials. If the other player is in role C, the players are opposed: only the score of the player in role A is increased if the player in role C wrongly identifies a concept; otherwise only the score of the player in role C is increased.

The quality of the input will be checked by the other roles, but the player with role A has an incentive to have at least some valid input to get a good score: as the player in role A does not know the role assumed by the other player, it must give correctly identified examples to win in the case where the other player assumes role B; and at least some complex and diverse examples in the case where the other player assumes role C.

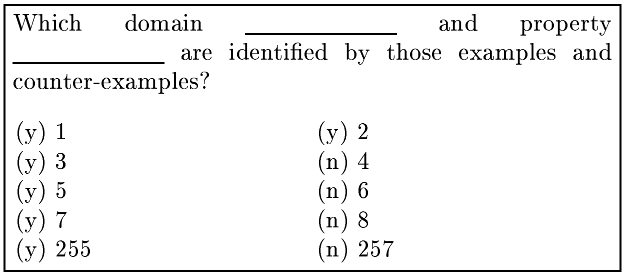
The computer assuming the role A knows whether the player assumes role B or C. If the player assumes the role C, the computer puts together some certified elements and some non-certified ones, asking the player to confirm or inform them. If no such data exists, then any human player should be assigned the role A to produce such a recording along with the data required. If the player assumes the role B, the computer replays a past recording of a human player assuming the role A. If no such role exists, then any human player should be assigned the role A to produce such a recording along with the data required.



Give 10 examples and counter-examples of `prime numbers` among `integers`:

(y) 1	(y) 2
(y) 3	(n) 4
(y) 5	(n) 6
(y) 7	(n) 8
(y) 255	(n) 257

Fig. 5. Role A: given a *domain* and a *property*, the player must enter 10 examples and counter-examples of the property in the domain.



Which domain ____________ and property ____________ are identified by those examples and counter-examples?

(y) 1	(y) 2
(y) 3	(n) 4
(y) 5	(n) 6
(y) 7	(n) 8
(y) 255	(n) 257

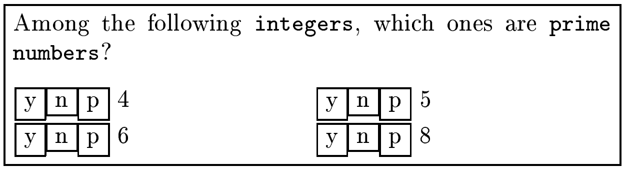
Fig. 6. Role B: given 10 examples and counter-examples of a property in a domain, the player must guess a *domain* and a *property*.

Role *B*: learn The player assuming role B must infer from 10 examples and counter-examples of objects satisfying a property the domain of all objects and the property considered. This role checks that the objects input by the player assuming role A are indeed answering the question: if not, the player assuming role B will never guess the domain and property. As there can still be several acceptable domains (e.g. “numbers”, “integers”), the player with role B is allowed five submissions, and signaled when any of the domain or property is correctly identified. The score of both players is increased when the player with role B correctly reproduces the domain and property, and stays the same otherwise.

Programming the computer to assume role B requires a bit more ingenuity: the computer obviously *knows* the domain and property, but must simulate a human behavior so that the player assuming the role A has an incentive to be truthful. For this purpose, the computer assuming role B must use the knowledge from previous games, with current and other domains and properties, to infer from the player’s example what the domain and property is. Data input by the player which has never been input before is ignored for this inference, but recorded to be later validated when the computer will assume role A.

Role *C*: MCQ The player assuming role C must simply answer correctly a multi-choice question quiz to identify among a set of objects which ones verify the property, in a minimum time. If the player with role C is incorrect, the score of the player with role A is increased. If the player with role C correctly identifies all the objects presented, the score increase is divided between the two players so that the player with role C gets more point if he answered quickly, and vice-versa.

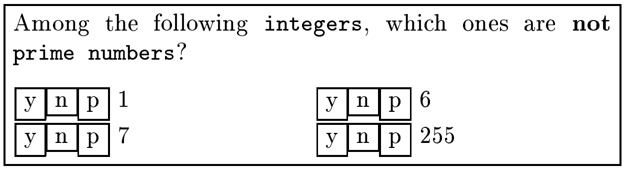
Programming the computer to assume role C is straightforward: the computer must and can only use the data previously validated for this domain and property. Examples which are not matched by previous data is “passed”, as a human would on an unknown example.



Among the following integers, which ones are prime numbers?

y n p 4 y n p 5
y n p 6 y n p 8

Fig. 7. Role *C*: given a domain, a property, and a random selection of objects in the domain, the player must correctly identify which objects verify the property.



Among the following integers, which ones are **not** prime numbers?

y n p 1 y n p 6
y n p 7 y n p 255

Fig. 8. Role *C*, variant: the player must correctly identify which objects *do not* verify the property.

Future Work

- In the current state, the valid pairs of "Domain" and "Property" values are entered by the administrator of the database, so that they stay in the context of the course taught. As those can be interconnected between questions (a property on a domain defines a sub-domain, on which further properties can be defined), it would be interesting to extend the system so that users can be oriented to enter their suggestion of properties in some domain, using additional roles and games to validate those entries.
- As the protocol was designed to minimize the burden on the user, it does not include any identification of the user from one session to the other. It would be interesting to add such a feature, first to memorize which concepts where previously validated and to be able to suggest more advanced topics, and second to "rank" the user by the reliability of their input, hence providing an additional incentive to provide input of high quality.

3 Comparison to Previous Results

3.1 Human Computation Principles

The two main general principles illustrated by human computation games such as `ESP` [2], `Peekaboom` [5] or `Verbosity` [4] are of symmetric and asymmetric verification game:

- In *asymmetric* verification game (illustrated by `Peekaboom` [5] and `Verbosity` [4]), the first user associates to an input x (e.g. a concept) some output $\{y_1, \ldots, y_k\}$ (e.g. properties of the concept), and the second user's interaction allows to *check* that this output indeed covers and is sufficient to describe the input. particular input is bounded. This technique is useful for cases where the number of inputs yielding a particular input is bounded.
- In *symmetric* verification game (illustrated by `ESP` [2]), one provides each user with a single question with many possible answers, and intersection the answer set of each user to identify the most pertinent one. This technique is useful for cases where the number of possible outputs for a particular input is bounded.

We used *combinations of asymmetric verification games* to extend the range of concepts managed by asymmetric verification games. This concept can be extended to other applications where the validity of an input cannot be checked by simple comparison to previous inputs. This is not possible with a single game (as `Peekaboom` or `Verbosity`) where the number of possible answers must be bounded to be able to check their validity.

The techniques illustrated by `VocaCIME` not only apply to the labeling of images in other languages than English, but also potentially applies to any other media such as sound or even video. The technique illustrated by CIMEQuiz, although similar to the approaches of Peekaboom and Verbosity [4] in the sense that it is based on an asymmetric confirmation game, is different in that it combines two complementary games to deal with the unlimited number of answers for player with role A: the role C serves to check the correction of those answers while role B serves to maintain a truthful incentive.

3.2 Bootstrapping Databases

We propose access protocols to databases which insure that the database grows in volume and quality proportionally to its usage. A similar concept was studied in *Mechanism Design*, where all players must contribute to a common goal to improve its worth for all, and where the mechanism must ensures that all players are pushed to contribute either in equal part, or in parts proportional to their benefit (generalizing the "Prisoner's dilemma's problem").

This concept can also be extended to more complex databases than those described for VocaCIME and CIMEQuiz, such as to databases where the diversity of the content of a database makes its quality, and where this diversity can be leveraged from the users in exchange of their access. Possible applications include databases of solved problems to be used in assignment [1], research papers (www.journal-ranking.com), amateur productions of music or video such as on UTube or Google Video.

It is to be noted though that the application of this concept to large user-submission will require more advanced techniques of quality control, and in particular the identification of the users by an account, to allow an expert to validate or invalidate an entry long time after the entry has been done.

Acknowledgment: The ideas presented in this report are a prolongation of many discussions with many different people over at least five years. A non-exhaustive list includes Julien Sebot, Joel Friedman, Jason Hartline, Kate Larson, Mike Terry, Matthias Schulze, Kasra Zokaei and Thomas Hazelton: thanks to all and to the others, potentially forgotten in this list.

Bibliography

[1] J. Barbay. On the use of a computer, to teach more and better, in a collective manner. In *X International Convention of Computer Sciences of Havana (INFORMATICA 2004)*, 2004.

[2] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, New York, NY, USA, 2004. ACM Press.

[3] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving accessibility of the web with a computer game. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 79–82, New York, NY, USA, 2006. ACM Press.

[4] L. von Ahn, M. Kedia, and M. Blum. Verbosity: a game for collecting common-sense facts. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 75–78, New York, NY, USA, 2006. ACM Press.

[5] L. von Ahn, R. Liu, and M. Blum. Peekaboom: a game for locating objects in images. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64, New York, NY, USA, 2006. ACM Press.