University of Alaska, Anchorage

CSCE A470

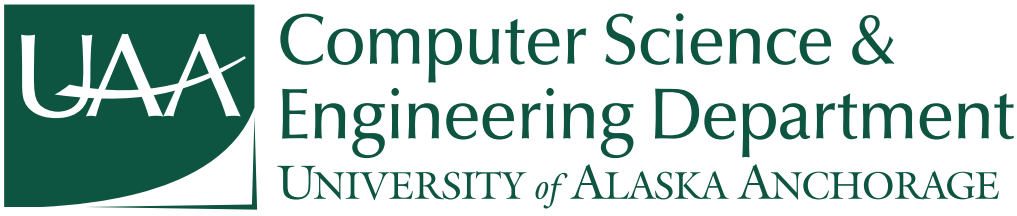
Capstone Project

A language-based CAPTCHA approach with application to semantic database construction

Author:
Teslin Roys

Supervisor:
Dr. Saif al Zahir

February 8, 2015



Abstract

In this paper, some theory and philosophy behind CAPTCHA (Completely Automated Public Turing Test to tell Computers and Humans Apart) systems is examined and a new language-based CAPTCHA system is described. The considered system provides the user with three phrases as options and asks the user to identify which phrases best construct a teleological meaning. The system has the ubiquitous application to securing websites from unwanted automated submissions, with potential additional value in improving semantic word maps similar to Princeton University's WordNet. Using a social feedback mechanism, this system may enable growth of a semantic map by periodically incorporating new randomly selected phrases which are identified to be contextually meaningful by users.

Keywords: CAPTCHA, semantics, syntax, language, AI, philosophy

1 Introduction

1.1 Motivation

A CAPTCHA, sometimes called a Human Interactive Proof, is a variation of the Turing test (Turing [1950]) in which a system authenticates users as human by verifying solutions to a problem which is in principle straightforward for humans to solve but difficult for current computers or algorithms. In general, preventing undesired (automated) use of Web services is an ap-

parent security need which CAPTCHAs attempt to satisfy. Mehra et al. [2011] argues that CAPTCHAs also present a viable means of mitigating the damage done by DoS (Denial of Service) attacks. There seems to be significant benefit in introducing new CAPTCHA techniques. As Von Ahn et al. [2003] suggest, the introduction of challenging AI problems for CAPTCHA mechanisms either leads to more secure systems or encourages advances in the field of AI. In addition, some CAPTCHAs in use today harness the problem-solving abilities of users to generate external benefits. For example, the reCAPTCHA system described by Von Ahn et al. [2008] uses results from users to help digitize books.

1.2 Organization

The remainder of this paper will discuss three main topics: (1) the aims of a CAPTCHA system in general, (2) comparison of several distinct CAPTCHA approaches and (3) give a preliminary description of a new language-based CAPTCHA system under development. This system will attempt to offer similar dividends that reCAPTCHA provides in book digitization to the domain of building semantic databases of words. We will discuss some of the tradeoffs of the proposed method and an overview of its design.

Figure 1: A prompt from the reCAPTCHA system.

2 Aims of a CAPTCHA system

2.1 Security features

It seems that a CAPTCHA system should ideally be efficient with respect to the time spent to generate a problem relative to the time spent by an attacker to solve it. In other words, there must seemingly be an asymmetry in the capability of algorithms to generate problem cases with solutions, and the capability of algorithms to solve an arbitrary case. This would appear particularly true when the goal is to stymie DoS-style (Denial of Service) abuse, as Mehra et al. [2011], because the aim may be to impose a computational cost on the attacker.

In addition, there must be an asymmetry between the algorithmic solver and a human user's ability to concoct a solution. This is apparently the case with OCR (optical character recognition) based systems, where the distorting transformations of typical CAPTCHA methods are more difficult to arbitrarily reverse than to apply. Nonetheless, it seems as though humans can find such a reversal trivial.

2.2 Usability

In order to gain any traction as a practical solution, we must expect that a provided CAPTCHA problem be not only less difficult for a human to solve than a computer, but require little time or effort on the part of the user. In addition, we would hope that the system's interface is readily accessible

and understandable by most human users. A balance must therefore be struck between complication of the scheme in order to befuddle machines, and streamlining to ensure ease of use by humans. The hope, of course, is that these are at least somewhat orthogonal goals.

2.3 Philosophy

It further reflects on our goals for any CAPTCHA system to hypothesize about a CAPTCHA which could sort humans and machines with perfect certainty. It is therefore reasonable to make a foray into the philosophical to ask whether one can set such a task which a machine finds not only difficult, but impossible, so that a machine will never be able to compose an answer. Of course, even hypothetically we are not simply defining a completely impossible task – to the contrary, a human must still be able to accomplish it. To suppose such a task to exist, we must suppose some unique and qualitative (rather than quantitative) difference between human and machine intelligence in principle. In other words, it posits a semantic gap (in the sense Von Ahn et al. [2003] describe it) which is in fact not incremental in nature but dramatically abrupt.

The existence of this kind of inherent qualitative difference between mind and machine is proposed by multiple authors including Penrose [1999] and Searle [1980]. The most resilient such claims frequently entail an appeal to the problem of syntax versus semantics. In some way, an objection to the principle of machine intelligence is usually offered on the grounds that a

machine cannot perceive semantics. But a hidden premise is also that the distinction between syntax and semantics is rigorous enough to allow us to draw such a line.

Penrose's argument is particularly interesting from a computational logic point of view, deriving from the idea of Gödel sentences. Gödel [1931] showed that given an axiomatic language of sufficient complexity, there will be true sentences in the language which cannot be proven. It is known from Turing [1936] that equivalently some problems do not have a decision procedure (undecidable problems). It is from our seeming ability to construct these Gödel sentences (or similarly, concoct undecidable problems) that Penrose argues we humans have special capabilities that machines do not.

However, this argument alone seems less than convincing. As McCarthy [1990] points out, it is possible to create a program in LISP which can construct Gödel sentences of its own once provided with some confidence independent of the formal derivation system. The question, then, is where human confidence in semantic truth comes from and if it is fully transferable or replicable. Presumably, if a machine were supplied with its own such confidence, it could pass Penrose's bar.

Other arguments against the possibility of human-indistinguishable machine intelligence run along similar syntax versus semantics lines. For example, Putnam [1980] makes the case that the Löwenheim-Skolem theorem of set theory can be applied to show how syntax underdetermines semantics. But even this is, again, a system-dependent argument. As Putnam says, the

Löwenheim-Skolem theorem demands the Zermelo-Fraenkel (ZF) axioms to hold, not to mention the Axiom of Choice (ZFC). This is debateably using insufficiently fundamental tools to answer a more fundamental question.

Regardless, if reasonable doubt exists whether there is a unique qualitative distinction between minds and machines, it supports a theory that CAPTCHAs should aim to emphasize relative improvement in the success rates of humans over machines rather than absolute accuracy. Not only may such harsh conditions of merit be practically infeasible, it is also unclear if there is theoretical basis for such a position.

3 Types of existing approaches

3.1 Visual OCR-based CAPTCHAs

A significant number of existing CAPTCHA schemes in use appear to involve variations or extensions of the concept of forcing the user to solve an optical character recognition (OCR) problem. These schemes are frequently referred to as text-based, but they are typified by their addition of visual noise to make recognition more difficult. As Baird and Riopka [2005] notes, a common attack on these type of schemes is based upon segmenting and identifying individual characters in the text.

Mori and Malik [2003] indicate that the popular Gimpy and EZ-Gimpy visual CAPTCHA implementations admit an automated solution in excess of 33% of the time, and advise that this makes the mechanisms an unreliable

defense. However, as discussed above, it is possible that the effectiveness of an implementation might be better measured by the computational cost imposed on an attacker rather than its ability to thwart all such attacks. Nonetheless, this is certainly an indication that there are improvements to be made over the strategy employed by these systems.

The handwriting based system Rusu and Govindaraju [2004] propose is perhaps such an improvement. Rusu and Govindaraju [2004] argue that natural handwritten text offers a greater challenge for word/letter segmentation and character recognition than conventional automated distortions and filters (e.g. like those used by Gimpy).

3.2 Language-based or mixed approaches

A wide variety of alternate methods have been proposed. Some, like the purely semantic and language-based EGGLUE system considered by Hernandez-Castro et al. are implemented by a sizeable number of websites. The EGGLUE system has a couple of notable usability advantages. Namely, it is better accessible for blind or otherwise disabled users via publically available text-to-speech software than typical OCR-based systems, and the constraints on the answer (must be an English verb) make it potentially less frustrating to pass for the average reasonably literate English language user.

Hernandez-Castro et. al give a good account of the faults of the EGGLUE implementation, including a reliance on common verbs and a general susceptibility to search-engine based attacks. The system appears to rely on a

static teleological database and the solution always consists of a verb, which makes it particularly vulnerable to a dictionary attack. Though the authors make a very good case about the flaws of the implementation, their broader conclusions about the viability of logic- or semantics-based CAPTCHAs and their recommendations regarding retreat from this general research direction seem less well founded. A general appeal is made to improving data mining techniques rendering semantics-based systems easily bypassed, but without a more substantial argument, the counterexample with regard to optical CAPTCHAs of steadily improving OCR techniques is easily deployed.

Systems can employ a multi-modal approach, as Almazyad et al. [2011] suggests, relying on a combination of simultaneous CAPTCHA mechanisms. This seems reasonable, as an attacker must then uniquely develop a solver which addresses a potentially unique combination of layered mechanisms. In general, we might tend to regard multi-factor authentication as being more secure. For this reason, ATM cards typically combine a token with a secret (PIN).

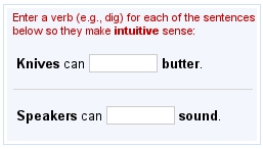


Figure 2: A prompt for the EGGLUE system.

Examples of multi-modal CAPTCHAs are prevalent, such as SemCAPTCHA (Łupkowski and Urbanski [2008]) which uses two techniques: (1) an "odd one out" approach where several possible Polish word answers are given and a semantic connection exists between all but one answer, and (2) these answers are displayed in an image hidden behind an OCR-challenging image distor-

tion. Notably, the example implementation of SemCAPTCHA relies on a small and static semantic database based on animal taxonomy (mammals, reptiles, etc.).

Visually distorting transformations are not the only technique to confuse automated readers. Misspelling, in-sentence word jumbling and abbreviation are a few kinds of basic transformations language-based CAPTCHAs might use to obscure words from identification by machine. A word may be misspelled by randomly altering a number of its letters to form a different but alphabetically similar word, and a human reader will frequently nonetheless be able to correctly identify the word from context. Likewise, a few words and so on may be rearranged or swapped in a sentence without unduly harming a human's ability to correctly determine its semantics (out of a list of grammatical sentences a human might identify the one with the most coherent meaning). Abbreviation or shorthand is another method of obscuration, perhaps especially interesting because it seems to employs some semantic principles. For example, the phrase "I see you" can be abbreviated to the phrase "i c u" by phonetic analogy.

3.3 Other approaches

Gossweiler et al. [2009] offers a computer vision based system which does not depend on OCR. This system asks the user to submit a correct orientation (e.g. what is "up") for a displayed image. Photographs are largely assumed to be taken in an upright position, but the authors do not entirely rely

upon this, as they expect their social feedback mechanism to correct for these kind of differences. Images of easily-detected objects are pruned from the database based on success and prevalence of specific object detection algorithms (e.g. images containing faces are omitted from selection), and images which are difficult for humans to orient are also removed via the social feedback mechanism. They identify images which are difficult for humans to orient upright by evaluating the average and deviation in angles that multiple users submit for a single candidate image. The presumption is that images with high variation in submitted angle are difficult to orient due to the lack of consensus.

4 Preliminary description of proposed method

4.1 Design

The proposed system involves a server and client component. The server authenticates users as human by issuing problems to solve at the request of the client. If the user solves sufficiently many (ideally in most cases only one) problem cases successfully they are allowed access to the web service.

4.2 Basis (match) phrases

Princeton University's WordNet (Miller [1995] is a lexical database which stores extensive relational semantic information. The database encodes words

as synonym-sets (or synsets) or groups of words with identical meaning. More information such as hyperonymy (super-subordinate relations) is also included, and an additional WordNet database stores teleological (purpose) information about words. For example, an audience is a (typical) "beneficiary" of showing a movie, which is encoded with the beneficiary relation.

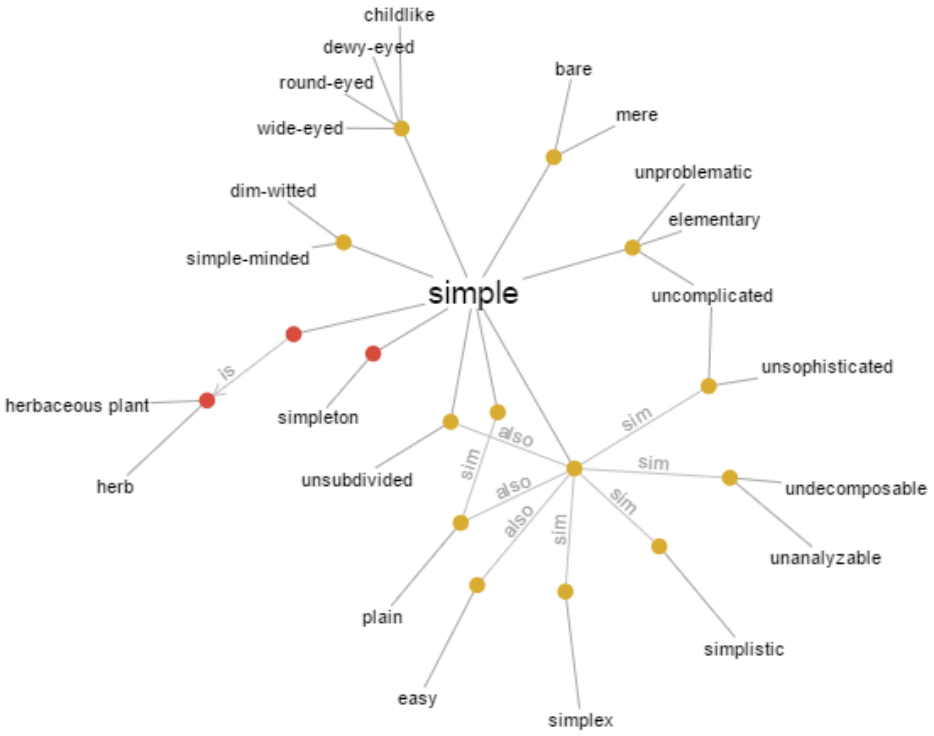


Figure 3: A visualisation of a small part of the WordNet database, produced by the WordVis software interface developed at the Norwegian University of Science and Technology Trondheim by Steven Vercruysse.

There are many ways to retrieve or construct meaningful phrases from

WordNet. An example of a meaningful teleological phrase is "woman drives car". Each synset in the database has various usage examples which usually consist of short phrases or sentences.

For our initial testing purposes, we may select a synsets at random and extract trigrams from some of their usage examples. We may then employ these trigrams as the basis of our pool of "match" phrases. However, using such a scheme may achieve subpar results if a hypothetical attacker has access to the same initial database. It would be preferable, then, to demonstrate a viable method of constructing meaningful phrases independently, e.g. using a noun-verb-noun structure where the words are selected to have a high degree of semantic closeness.

4.3 Problem structure

A problem is presented to the user to identify which of the three phrases provided are most meaningful. The user is provided with three such phrases, with one being a constructed phrase with known meaning and the other two being randomized or mutated examples. The problem, then, is a 3-tuple (M, C, R) consisting of the three phrases, with each phrase consisting of a 3-tuple in turn of two nouns and a verb in a certain order.

In the scheme, one of the three example phrases will be the known meaningful match M. One of the unknown examples will be a generated candidate C. The other, R, will be randomized and presumed incorrect unless the user is already authenticated as human. M and C can be arbitrarily selected from

a pool of match and candidate phrases respectively.

This simplistic problem structure is presented with marginal loss of generality, out of a desire to keep a proof-of-concept system minimalist rather than because of methodological constraints. It is not difficult to envision a more sophisticated implementation using longer or more varied phrase structure with less dependence on nouns.

4.4 User choice mechanism

A user is presented with a triangular area with each vertex of the triangle labeled with one of three phrases. The user is prompted to click the cursor within the triangle nearest the one or two phrases whose 'action makes the most sense'. A certainty score is displayed next to each vertex based on where the cursor is pointed. The certainty score is inversely proportional to the distance of the cursor to the vertex in the following way:

$$C_n = \frac{m}{max(d, 1)}$$

where m is the maximum distance allowed from the vertex and d is the selected distance. A user's choice can be represented as a 3-tuple m, c, r of certainty scores taken for each vertex.

4.5 Passing/failing and multiple attempts

The user's choice is given in terms of degree of certainty, so too should the acceptance/non-acceptance be based upon a threshold of certainty as well.

The 3-tuple (m, c, r) of the user choice is examined with reference to the matched phrase, the candidate phrase, and the random phrase. Let $Q_1, Q_2, \ldots, Q_n$ be the quality rating of a sequence of successive choices by the user. The degree of quality of the choice numbered i is given by the following:

$$Q_i = m_i - r_i$$

If the $\sum_i^n Q_i$ exceeds some positive threshold T_1 then the user passes. An insufficiently high Q can result in one of two consequences. First, the user may be given another choice to make if $\sum_i^n Q_i$ is positive. Second, they may be banned for further trial if $\sum_i^n Q_i$ negatively exceeds some negative threshold T_2.

4.6 Candidate phrase feedback

A candidate phrase can become a match phrase if a consensus of passing users forms. The acceptance of a candidate phrase C is contingent on the average of the certainty scores A of passing users who were given the same problem (M, C, R). If A exceeds some threshold T_3 the candidate becomes a match. Meanwhile, if A negatively exceeds a negative threshold T_3 the

candidate may be removed as an anomaly.

4.7 Candidate formation

Candidate phrases may be formed in two ways. Firstly, a random phrase may be promoted to the candidate pool if a passing user has certainty r exceed some threshold T_5.

Secondly, to introduce diversity into the system, high certainty match phrases can be periodically mutated by a random swap of one noun or the verb in the phrase, and added back to the candidate pool.

5 Observations and concerns

5.1 Usability

The system must be evaluated on grounds of usability, particularly whether the prompt can be easily understood by users. The user's grasp of the graphical layout and meaning of their choice is as essential to authentication as feedback. In addition, a worthwhile avenue of study may be whether dyslexic individuals find the tasks unreasonably challenging.

Finally, it will require testing to determine if stochastically generated three word phrases will too frequently have meaning for users, causing confusion by generating problems with only meaningful phrases. This could be mitigated by choosing longer phrases, seemingly decreasing the probability

that a given set of randomly selected words will have a coherent meaning.

5.2 Security

One primary security concern is increasing the difficulty of a dictionary attack on the system. The sensitivity of the responses to repeated wrong answers seemingly precludes a blind dictionary attack, but some valid worry remains regarding more sophisticated approaches. For example, an attack based on search engine query results may be effective.

Combination of the considered method with a traditional OCR approach employing visual distortion on the displayed phrases may reduce accessibility for blind persons, for example, but demand a more holistic attack.

5.3 Further study

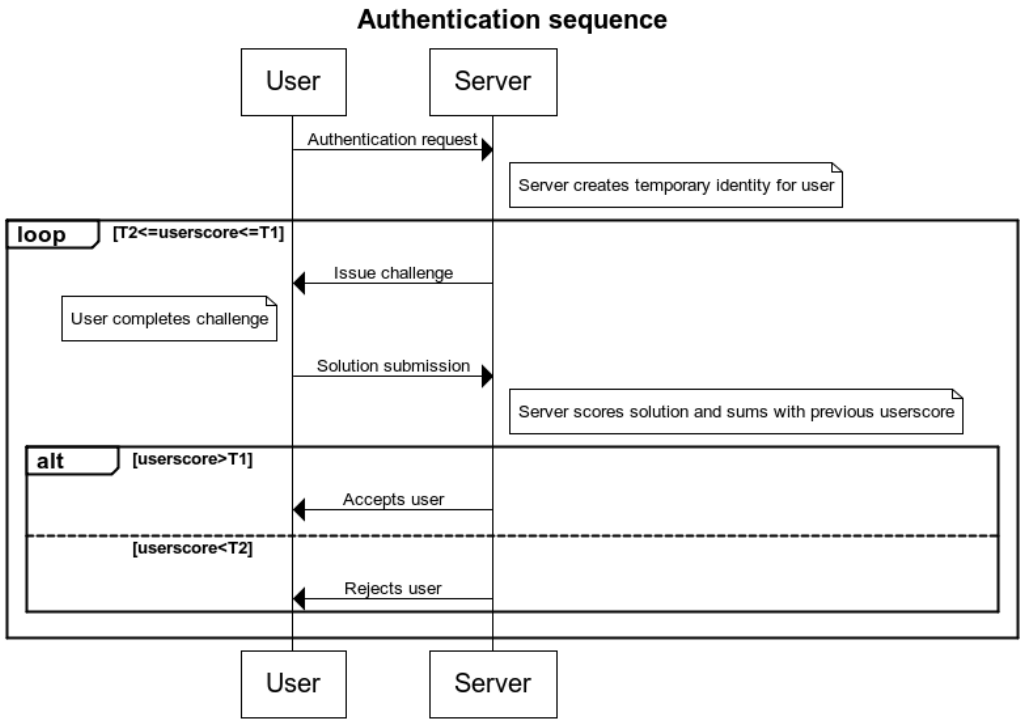
The system does not identify a type of teleological relationship between words in a phrase match, as WordNet's data does. Extensions of the concept using a similar multilevel, triangular certainty scheme could work to identify the relational type of a phrase. This would be at a tradeoff of additional prompts for users, which would ideally be generally avoided.

Additionally, study may be required to find adequate threshold values for $T_1 \ldots T_5$ based on various requirements of phrase pool diversity, user reaction to the sensitivity level, and the quality and rate of accumulating match phrases. However, like many aspects of this design, proper estimation of

these thresholds might require at least a limited deployment of a complete software package to some minimal userbase.

Overall, the usefulness of the method should be evaluated on the merits of its usability and security but secondarily upon the quality of its new semantic data acquisitions.

6 Appendix A: Preliminary UML



7 Appendix B: Preliminary code

The following code can also be found on GitHub at this address:

`https://github.com/teslinroys/wc_captcha`

It is published under the MIT License, see LICENSE file for details. For code documentation, TypeDoc is used for the TypeScript code and Docco for the Node.js code.

7.1 Clientside (Typescript/HTML5)

```
1  class Vector2 {
2      public X: number;
3      public Y: number;
4       /**
5   * Represents a 2D vector.
6   */
7      constructor(x: number, y: number) {
8          this.X = x;
9          this.Y = y;
10     }
11     /** Returns the distance between this vector and the input
           vector. */
12     distanceTo(v2: Vector2): number {
13         var a = this.X - v2.X;
14         var b = this.Y - v2.Y;
15         var c2 = Math.pow(a, 2) + Math.pow(b, 2);
```

```
16          return Math.sqrt(c2);
17      }
18  }
19
20  class Captcha {
21      public con: CanvasRenderingContext2D;
22      public canvas: HTMLCanvasElement;
23      public control_pts:Vector2[];
24   /**
25   * Constructs the interactive CAPTCHA element.
26   */
27      constructor(c: HTMLCanvasElement) {
28          this.con = c.getContext('2d');
29          this.canvas = c;
30          this.canvas.addEventListener("mousedown", (event:
                MouseEvent) => this.onMouseDown(event), false);
31          this.control_pts = [new Vector2(250, 50), new Vector2
                (50, 450), new Vector2(450, 450)];
32      }
33      /** This event handler redraws the canvas when it is clicked
            . */
34      onMouseDown(e: MouseEvent) {
35          var x, y;
36          if (e.pageX || e.pageY) {
37              x = e.pageX;
38              y = e.pageY;
39          }
```

```
40          else {
41              x = e.clientX + document.body.scrollLeft +
42              document.documentElement.scrollLeft;
43              y = e.clientY + document.body.scrollTop +
44              document.documentElement.scrollTop;
45          }
46          // Convert to coordinates relative to the convas
47          x -= this.con.canvas.offsetLeft;
48          y -= this.con.canvas.offsetTop;
49
50          var mp: Vector2 = new Vector2(x, y);
51          var m: Vector2 = this.control_pts[0];
52          var c: Vector2 = this.control_pts[1];
53          var r: Vector2 = this.control_pts[2];
54          var b: boolean = this.pointInTriangle(mp, m, c, r);
55          this.draw();
56          if (b == true) {
57              this.con.fillText("m=" + Math.round(mp.distanceTo(m)
                    ) +" c=" + Math.round(mp.distanceTo(c)) + " r=" +
                     Math.round(mp.distanceTo(r)), 50, 50, 80);
58              this.con.beginPath();
59              this.con.moveTo(m.X, m.Y);
60              this.con.lineTo(mp.X, mp.Y);
61              this.con.moveTo(c.X, c.Y);
62              this.con.lineTo(mp.X, mp.Y);
63              this.con.moveTo(r.X, r.Y);
64              this.con.lineTo(mp.X, mp.Y);
```

```
65              this.con.closePath();
66              this.con.strokeStyle = 'rgb(32, 128, 64)';
67              this.con.stroke();
68
69          }
70      }
71      /** Draws the CAPTCHA element to the specified canvas. */
72      draw() {
73          this.con.clearRect(0, 0, this.con.canvas.width, this.con
                .canvas.height);
74          this.con.beginPath();
75          this.con.moveTo(this.control_pts[0].X, this.control_pts
                [0].Y);
76          for (var i = 0; i < this.control_pts.length; i++) {
77              this.con.lineTo(this.control_pts[i].X, this.
                    control_pts[i].Y);
78          }
79          this.con.lineTo(this.control_pts[0].X, this.control_pts
                [0].Y);
80          this.con.closePath();
81          this.con.strokeStyle = 'black';
82          this.con.stroke();
83      }
84      /** Returns the cursor position relative to the canvas. */
85      getCursorPosition(e) {
86      var x;
87      var y;
```

```
88      if (e.pageX || e.pageY) {
89          x = e.pageX;
90          y = e.pageY;
91      }
92      else {
93          x = e.clientX + document.body.scrollLeft +
94          document.documentElement.scrollLeft;
95          y = e.clientY + document.body.scrollTop +
96          document.documentElement.scrollTop;
97      }
98      // Convert to coordinates relative to the convas
99      x -= this.con.canvas.offsetLeft;
100     y -= this.con.canvas.offsetTop;
101
102     return [x, y]
103 }
104
105     /** Returns whether or not a given point (e.g. mouse
            position) is inside a triangular area. */
106     pointInTriangle (point:Vector2, v1:Vector2, v2:Vector2, v3:
            Vector2) : boolean {
107         var A = (-v2.Y * v3.X + v1.Y * (-v2.X + v3.X) + v1.X * (
                v2.Y - v3.Y) + v2.X * v3.Y) / 2;
108         var sign = A < 0 ? -1 : 1;
109         var s = (v1.Y * v3.X - v1.X * v3.Y + (v3.Y - v1.Y) *
                point.X + (v1.X - v3.X) * point.Y) * sign;
```

```
110             var t = (v1.X * v2.Y - v1.Y * v2.X + (v1.Y - v2.Y) *
                    point.X + (v2.X - v1.X) * point.Y) * sign;
111             return s > 0 && t > 0 && s + t < 2 * A * sign;
112         }
113 }
114
115 var captcha;
116
117 /** Loads CAPTCHA element. */
118 window.onload = () => {
119     var c = <HTMLCanvasElement> document.getElementById('captcha
            ');
120     captcha = new Captcha(c);
121     captcha.draw();
122 };
```

7.2 Serverside (Node.js)

```
1 //First load HTTP functions.
2 var http = require('http');
3 //We will use the port defined in environment, otherwise 1337.
4 var port = process.env.port || 1337;
5
6 //Create an HTTP server to listen for clients.
7 http.createServer(function (req, res) {
8     res.writeHead(200, { 'Content-Type': 'text/plain' });
9     res.end('Hello user\n');
```

```
10      //Define a wordnet object. Then use it to retrieve
            information about a synset.
11      var wordnet = new WordNet();
12      wordnet.lookup('simple', function (results) {
13          results.forEach(function (result) {
14              console.log('------------------------------------');
15              console.log(result.synsetOffset);
16              console.log(result.pos);
17              console.log(result.lemma);
18              console.log(result.synonyms);
19              console.log(result.pos);
20              console.log(result.gloss);
21          })
22      });
23  }).listen(port);
```

References

Abdulaziz S Almazyad, Yasir Ahmad, and Shouket Ahmad Kouchay. Multimodal captcha: A user verification scheme. In *Information Science and Applications (ICISA), 2011 International Conference on*, pages 1–7. IEEE, 2011.

Henry S Baird and Terry P Riopka. Scattertype: a reading captcha resistant to segmentation attack. In *Electronic Imaging 2005*, pages 197–207. International Society for Optics and Photonics, 2005.

Elie Bursztein, Matthieu Martin, and John Mitchell. Text-based captcha strengths and weaknesses. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 125–138. ACM, 2011.

Monica Chew and J Doug Tygar. Image recognition captchas. *Information Security*, pages 268–279, 2004.

Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38(1): 173–198, 1931.

Rich Gossweiler, Maryam Kamvar, and Shumeet Baluja. What's up captcha?: a captcha based on image orientation. In *Proceedings of the 18th international conference on World wide web*, pages 841–850. ACM, 2009.

Carlos Javier Hernandez-Castro, Arturo Ribagorda, and Julio Cesar Hernandez-Castro. On the strength of egglue.

David Hilbert and Wilhelm Ackermann. Grundzüge der theoretischen logik. *Berlin, Heidelberg*, 1928.

Graeme Hirst and Alexander Budanitsky. Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering*, 11(01):87–111, 2005.

Paweł Łupkowski and Mariusz Urbanski. Semcaptcha—user-friendly alternative for ocr-based captcha systems. *Speech and Language Technology*, 11:278–289, 2008.

John McCarthy. Review of the emperor's new mind by roger penrose. *Bulletin of the American Mathematical Society*, 23(2):606–616, 1990.

M Mehra, M Agarwal, R Pawar, and D Shah. Mitigating denial of service attack using captcha mechanism. In *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, pages 284–287. ACM, 2011.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual captcha. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–134. IEEE, 2003.

Roger Penrose. *The emperor's new mind: concerning computers, minds, and the laws of physics*. Oxford University Press, 1999.

Hilary Putnam. Models and reality. *The Journal of Symbolic Logic*, 45(03): 464–482, 1980.

Narges Roshanbin and James Miller. A survey and analysis of current captcha approaches. *Journal of Web Engineering*, 12(1-2):1–40, 2013.

Amalia Rusu and Venu Govindaraju. Handwritten captcha: Using the difference in the abilities of humans and machines in reading handwritten words. In *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*, pages 226–231. IEEE, 2004.

John R Searle. Minds, brains, and programs. *Behavioral and brain sciences*, 3(03):417–424, 1980.

Alan M Turing. Computing machinery and intelligence. *Mind*, pages 433–460, 1950.

Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. *J. of Math*, 58:345–363, 1936.

Luis Von Ahn, Manuel Blum, Nicholas J Hopper, and John Langford. Captcha: Using hard ai problems for security. *Advances in Cryptology—EUROCRYPT 2003*, pages 294–311, 2003.

Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

Amber Wilcox-O'Hearn, Graeme Hirst, and Alexander Budanitsky. Real-word spelling correction with trigrams: A reconsideration of the mays, damerau, and mercer model. *Computational Linguistics and Intelligent Text Processing*, pages 605–616, 2008.

Rong Zhao and William I Grosky. Negotiating the semantic gap: from feature maps to semantic landscapes. *Pattern Recognition*, 35(3):593–600, 2002.