TURING TEST, EASY TO PASS; HUMAN MIND, HARD TO UNDERSTAND

CRISTI STOICA

Abstract. Under general assumptions, the Turing test can be easily passed by an appropriate algorithm. I show that for any test satisfying several general conditions, we can construct an algorithm that can pass that test, hence, any operational definition is easy to fulfill. I suggest a test complementary to Turing's test, which will measure our understanding of the human mind. The Turing test is required to fix the operational specifications of the algorithm under test; under this constrain, the additional test simply consists in measuring the length of the algorithm.

Contents

1. Passing the Turing test – A brute force solution

1.1. The Turing test

Turing [1] replaced the question "can machine think?" with a more testable question: "can machine simulate the human thinking, in such way that a human judge cannot distinguish that machine from a human?" In fact, it is not so much a problem of the machine. Of course, the storage capacity and power of computing will increase the chances, but the problem is mainly about the software running on that machine.

Turing also predicted that in about 50 years there will exist machines able to appear as humans to 30% of human judges, and that more and more people will believe that it is possible for machnies to think, according to this definition. And, indeed, more and more machines are programmed and trained to pass the Turing test, and they give more

Email: holotronix@gmail.com.

and more human-like answers. Therefore, it may be late to add something to Turing's prediction.

I will propose an argument, which has two purposes. One purpose is to support Turing's prediction, that someday we will be able to build "thinking machines". The second purpose is to show that building such machines is not that difficult. Of course, Turing knew this, and for him it was very clear, although his prediction was, at that time, very disputed. Now, we accept much easily this possibility (as he predicted), because we saw what a computer, running an appropriate software, is able to do. Of course, there are voices complaining about how dehumanized we become, because of the rise of the machines. Listening to those voices, we may think that Turing's test will be passed by the machines just because we will become more like them, rather than them becoming more human-like.

1.2. Algorithmic tests

The Turing test is a way to deal with the mind problem in an operational manner. We consider two black boxes, one containing the human, the other containing the computer. A judge communicates with them by inputing text – exactly like we are doing when we chat by using a messenger application – with the purpose of identifying the human. We can consider that the test consists in a list of questions, and for each question a range of answers are considered as valid, that is, they may be given by a human being. We can even decide, in function of the received answer, what is the next question. To sum up, I assume that the Turing test is an *algorithmic test*:

Definition 1.1. An *algorithmic test* is a test that satisfies the following set of conditions:

(1) The questions and answers are written in an alphabet having a finite number N of symbols.
(2) Each question is a text of finite length.
(3) Each answer is expected to have a finite length. Optionally, the questions and answers may have attached to the text pictures, movies, audio, or other type of files, which are required to be in a digital format, and of finite size.
(4) The number of the questions in the test is finite.
(5) The judge can even repeat questions, to detect some changes in the answers (humans change their minds often), but only a finite number of times.
(6) The judge is allowed to decide what question will follow, based on the previous answers (from a finite list of items, according to item (4)).
(7) Each question admits at least a valid answer.
(8) For each question, there is an algorithm taking the answer as input and returning "YES" if the answer is accepted, and "NO" otherwise.
(9) First time when an answer is evaluated with "NO", the testing algorithm stops, with the output "NO"; otherwise, when the list of questions is ended, the testing algorithm stops with the output "YES".

The conditions (1)-(6) above are very pertinent. In fact, it is hard to imagine how somebody can even break them. I think it is safe to assume that those conditions don't reduce too much the generality of the Turing test, since:

- No judge is able to ask questions of infinite length, or conduct an interrogatory by asking an infinite number of questions.
- No human is expected to deal with such infinite interrogatories.
- No messenger program can stream an infinite amount of bytes.
- No computer can process an infinite amount of bytes.

The condition (8) may seem more restrictive. Turing transformed the vague question "can machine think?" into a well stated criterion. But, this criterion leaves at the latitude of the judge – presumably human – what questions to ask. The range of possibilities seems unlimited. To see that the condition (8) is not that restrictive as it may look, let's consider a test satisfying the conditions (1)-(6) (and not necessarily 8), with the condition (3) replaced by

(3') The answers are allowed to have a (finite) maximum length N_A.

This condition is not very limiting, because we can fix any length we want. For example, the length of the text in a book, or the size of a CD, or a DVD, or of the largest storage device you can imagine, will satisfy all practical purposes. The condition (3') implies the condition (8). To see this, let's list all the possible texts of maximum length N_A – there are "only" $\displaystyle\sum_{l=1}^{N_A} N^l = \frac{N^{N_A+1}-1}{N-1} - 1 < \infty$ such answers. Some of them are valid, and some not, and we can always create an algorithm that simply associates to each valid answer a boolean value "YES", and to each invalid one "NO". Therefore, the implication follows. This means that the condition (8) is not that restrictive. Even if the algorithm that evaluates the answer depends on the previous questions and answers, it will still be an algorithm.

1.3. **The brute force solution**

Since the judge is limited to a finite length test, the test can be written down in a computer. The test is nothing but an algorithm (a Turing machine) that decides, based on some input, in a finite number of steps, an answer between "YES" and "NO". We can program a computer to play the role of the judge. It is simple; every computer programmer can do this, starting from the list of questions. Therefore, the judge can be a machine.

It is an easy task to construct an algorithm that passes an algorithmic test. All it has to do that algorithm, is to generate all possible answers of length n, apply itself the test to each of them, until it finds a valid answer, and then return that answer. If it does not find a valid answer, it simply goes to $n + 1$ and repeats the validation for each answer. If it starts with $n = 1$, then, after a probably large but finite number of trials, it will find a valid answer. We conclude that

Lemma 1.2. For each algorithmic test there exists an algorithm that passes that test.

It follows that any Turing test which is algorithmic can be passed by a brute force algorithm like this one.

2. Understanding the human mind

2.1. How to win by using the brute force

The previous section showed that the Turing test can be easily passed. So, Turing was right. His test can be passed, but we saw so far that it can be passed by a brute force algorithm. Does this result trivialize the whole idea of the Turing test? Well, it depends. If we put the entire weight on just passing the test, then this result shows that any properly defined test is easy to pass.

The brute force approach has a large difficulty. We have to collect a large number of questions and validation algorithms, and to feed them to the algorithm we want to "train". From a practical viewpoint, we can do the followings.

(1) Make an online database with questions relevant for the Turing test.
(2) Ask people to contribute online to the database.
(3) Implement the algorithm that reads the test, and for each question generates a valid answer.
(4) Ask people to test online the algorithm.

I expected that this can be done without problems.

2.2. It is not important to win, but how you win

I want to suggest that the central point is not to pass the Turing test. The central point is to find better algorithms that do this. It is not important to win, but how you win.

An example from physics will emphasize the idea. If we throw a rock, the Earth's gravity force will make it follow a parabolic trajectory. Suppose we have an enumeration of all possible ways to throw the rock, depending on the launching point, and the launching direction and velocity. This enumeration is not discrete, since the parameters vary continuously, but we just assume that we have discretized them appropriately, and we obtain something like an old fashion artillery table. We can construct a brute force algorithm that stores such a table, and for each initial conditions returns the corresponding trajectory, as it is recorded in the table. But, we can use a much better algorithm, one based on Newton's laws. An appropriate algorithm can compute a trajectory from an equation, for any set of initial data. This second algorithm is much shorter than the brute force one. This shortness is due to the fact that we know and apply the laws of physics.

Similarly, if we know the laws governing the human mind, we can improve very much our mind algorithm. A large number of similar question-answer pairs will be replaced by a much smaller number of algorithms that calculate the answer, based on the mind's laws.

Suppose that we created the large database of questions and answers. Organizing them, finding common points, finding logical implications between the items of the Turing test, can be done by a better understanding of the way the human mind works. Finding the mind's laws is equivalent to finding the best compression algorithm for the

Turing test. Thus, the length of the algorithm simulating the human mind is a measure of our success in understanding this mind.

I suggest that the challenge is two-directional:

(1) Creating a Turing test. This automatically leads to at least one "thinking algorithm".
(2) Creating a "thinking algorithm" as short as possible, that passes that Turing test.

2.3. Testing our understanding of the mind

Let's consider two algorithms that pass the same Turing test. Then, the shortest algorithm from the two contains a better codification of the human mind. This reveals a better knowledge about how the mind works. Note that it is irrelevant to compare two algorithms based on their length only, important is to impose a constrain regarding what those algorithms do, and this constrain is provided by the Turing test.

This second test is intended to complement the Turing test. Since each Turing test can be passed by an appropriate algorithm, which can be easily constructed from the test itself, it follows that the operational view has some limits. Our progress in understanding a phenomenon is not measured only by our capacity to simulate that phenomenon. The simulation is specified in terms of inputs and outputs, like a black box. Understanding the laws governing the phenomena allows us making a shortest description of them, hence, a shortest implementation of that black box. It is this shortness the measure of our progress.

To summarize, the compression of a specific Turing test is a measure of our understanding of the human mind. Besides the Turing test, it is important our understanding of the mind, and this understanding can be encoded in how general are the rules we discovered.

References

[1] Turing, A.M., *Computing machinery and intelligence.* Mind, 59, 433-460, (1950).