Caden Kroonenberg
Problem Set 1
EECS 649
1-24-22

1.1 The Internet Objection

The Internet contains an estimated 1.18 billion websites. There is a vast amount of information found online – it can be thought of as mankind's collective knowledge. It is likely that answers to any question can be found through search engines (like Google). Objective questions (with a right or wrong answer) can surely found on the internet. Furthermore, answers to subjective questions (how do you feel about X, what is your opinion on Y, etc.) can be surmised through information which people have written about on the internet (through blogs, social media, articles, etc.). The machine would need to be designed in such a way as to not output contradicting answers to subjective questions, which would make for a much more believable imitation. This could be sufficient to win the imitation game, but because the machine simply synthesizes information from the internet, is it truly intelligent?

I personally find this objection (or rather the approach) interesting because it mimics how we as humans learn and grow to know things - for the most part. Unless one is discovering new information through research, it is typical to learn objective truths because someone else tells us it is true. We learn arithmetic through schooling where teachers tell us how different mathematical operators work in the same fashion as I learned programming in a college classroom (including an online classroom). We learn objective truths based on information that mankind has already found to be true. When it comes to opinions, it is not rare that people base their opinions on those of other people – I believe it would be fair to say a machine has certain opinions if it bases them on those of people. Furthermore, one could say a machine knows emotion if it mimics the typical human response to certain situations.

1.2 Are reflexes rational? Are they intelligent?

Reflexes are indeed rational. It would be hard to make an argument that they aren't. If your hand touches a hot surface, it would be rational to remove it as fast as possible (as a reflex) to avoid any damage to your hand. Reflexes, by definition, do not require thought to execute. This is a good thing – you wouldn't want to have your hand sitting on the stove for the extra portions of a second while you think to remove it. This also means that reflexes are not intelligent – they do not require thought, decision making, applying knowledge, etc.

1.3 A computer cannot be intelligent – they can only do what their programmers tell them. Is the latter statement true, and does it imply the former?

I do not think the latter statement is entirely true. In a Generative Adversarial Network (GAN), for example, the program synthesizes new creations which the programmer does not explicitly create. The programmer may choose what input data is used and how the weights and biases of the internal neural nets of the GAN affect the output, but it is still the machines which synthesizes the final product. The programmer would not know the final output due to randomness involved in the synthesis process.

It does not matter if the latter implies the former as the latter is false.

1.4 State of the Art in AI

a. Play a decent game of ping pong
   Yes, German researchers at the University of Tübingen used machine learning and robotics to construct a robotic arm which can play ping pong. It only took around an hour and a half of training (both simulated and physical) to learn how to play. The system tracks the location of the ball every 7 milliseconds. The algorithm uses this information as an input and outputs the optimal location of the paddle attached on the arm.

h. Write an intentionally funny story
   No. It is possible for AI to synthesize things which are funny like a stand-up comedy act, but this is only funny because the inputs to whatever algorithm produced this (in this case, 400,000 hours of stand-up comedy acts). It can make funny stories, but the intention to make the story funny is derived from the programmer behind the AI. It would require the AI to understand what humor and comedy is for it to write a funny story with the intention being the AI’s alone.

1.5 Cognitive model for two-summand addition

Sample input:	->	Sample output:
33, 105	->	138
6109, 463	->	6572
987345, 662871	->	1640216

```python
import time, random, sys
time.sleep(2 + random.random()) # Allow for 2-3 seconds to "type" first prompt
a_in = int(input("What is the first number?\n"))
a_digit_count = len(str(a_in))  # Number of digits in 'a'
a = []                          # Store a's digits in array

for i in range(a_digit_count):  # For each digit ...
    a.append(a_in % 10)         # Convert integer digit to array entry
    a_in//=10                   # Move to next digit
    # 0.1 - 0.25 seconds to process the digit, (mental note, write down the digit, etc.)
    time.sleep(0.1 + 0.15*random.random())

time.sleep(2 + random.random()) # Allow for 2-3 seconds to "type" second prompt
b_in = int(input("What is the second number?\n"))
b_digit_count = len(str(b_in))  # Number of digits in 'b'
b = []                          # Store b's digits in array

for i in range(b_digit_count):  # For each digit ...
```

```
    b.append(b_in % 10)         # Convert integer digit to array entry
    b_in//=10                   # Move to next digit
    time.sleep(0.1 + 0.15*random.random()) # Process the digit, (mental note, write down digit, etc.)

time.sleep(1 + random.random()) # Allow 1-2 seconds to type acknowledgement
print('Let\'s see', end='')
for i in range(random.randint(1,5)): # Add unnecessary variation
    print('.',end='')
print()

small_len, large_len = (a_digit_count, b_digit_count) if a_digit_count < b_digit_count else
(b_digit_count, a_digit_count)

c = [0] * large_len # Result array

for i in range(small_len):
    c[i] = (a[i] + b[i] + carry) % 10       # Denote one's digit of sum
    carry = a[i] + b[i] + carry >= 10       # Denote carry if necessary
    time.sleep(random.random()*0.25 + 0.25)  # Take 0.25-0.5 seconds for math operation
    # (0.5 * digit #)% chance that the carry is forgotten - Error more likely with larger numbers
    if carry and random.random() < (i+1)*0.005:
        carry = False
    elif carry: # If carry is remembered..
        time.sleep(random.random()*0.1 + 0.1) # .. take 0.1-0.2 seconds to process this

# If one summand is longer than the other
for i in range(small_len, large_len):       # For each extra digit the larger summand has ...
    to_add = a[i] if a_digit_count == large_len else b[i] # Add digit from larger summand
    c[i] = (to_add + carry) % 10            # Add digit to carry if necessary
    time.sleep(random.random()*0.05 + 0.25)  # Take 0.25-0.3 seconds for math operation
    carry = to_add + carry >= 10            # Denote carry if necessary
    if carry and random.random() < (i+1)*0.005: # (0.5 * digit #)% chance that the carry is forgotten
        carry = False
    elif carry: # If carry is remembered..
        time.sleep(random.random()*0.1 + 0.1) # .. take 0.1-0.2 seconds to process this

if carry: # Account for last carry
    c.append(1)

time.sleep(random.random()*0.5 + 0.15*len(c)) # Take 0.15 * number of digits +/- 0.5 seconds to type
result
print('The answer is ', end='')
for i in reversed(range(len(c))):           # Print largest digit first in output
    print(str(c[i]), end="")
print("")
print(':)') # Display human emotions :)
```

1.6 Shannon’s “Mind Reading” Machine

a. HHTHTTHTTHHTTTTTHHHHTHTTHHTTTTHH
b. Tosses 2-21: HTHTTHTTHHTTTTTHHHHT. P(Heads) = 9/20 = 45%
c. P(H|H) = 5/(5+5) = 5/10 = 0.5
   P(T|H) = 5/(5+5) = 5/10 = 0.5
   P(H|T) = 4/(4+6) = 4/10 = 0.4
   P(T|T) = 6/(4+6) = 6/10 = 0.6
d. HHHTHHTTTHTHTHHTHTTTTTHHTHHTTHHH
   Tosses 2-21: HHTHHTTTHTHTHHTHTTTTT. P(Heads) = 9/20 = 45%
   P(H|H) = 4/(4+6) = 4/10 = 0.4
   P(T|H) = 6/(4+6) = 6/10 = 0.6
   P(H|T) = 5/(5+5) = 5/10 = 0.5
   P(T|T) = 5/(5+5) = 5/10 = 0.5

e. Hardcoded input. Sample output:

```
Real Coin
T T F F F F F T F F
3 correct predictions   (30%)
7 incorrect predictions (70%)

Simulated Coin
T T F T F F F F T T
8 correct predictions   (80%)
2 incorrect predictions (20%)
```

Code:

```python
# Author: Caden Kroonenberg
import random
# Heads = True, Tails = False
def predict(actual, initial, HT_prob, TH_prob):
    correct = 0                         # Count # of correct predictions
    state = initial                     # State at toss 22 = Heads
    for i in range(10):                 # For 10 'tosses'
        if state:                       # If Prev state = Heads
            if random.random() <= HT_prob:  # HT_prob probability state changes (to tails)
                state = False
        else:                           # If Prev state = Tails
            if random.random() <= TH_prob:  # TH_prob probability state changes (to heads)
                state = True
        if state == actual[i]:     # Track correct predictions
            correct+=1
        if state:
            print('T', end=' ')
        else:
            print('F', end=' ')
    print()
    print(str(correct) + ' correct predictions\t(' + str(correct*10) + '%)') # Print correct results
    print(str(10-correct) + ' incorrect predictions\t(' + str((10-correct)*10) + '%)')  # Print error
    print()

# Real coin
real_actual = [False, False, True, True, False, False, False, False, True, True] # Actual tosses 23-32
real_initial = True # Toss 22 result
real_HT_prob = 0.5  # P(T|H)
real_TH_prob = 0.4  # P(H|T)
print('Real Coin')
predict(real_actual, real_initial, real_HT_prob, real_TH_prob)

# Simulated coin
sim_actual = [True, True, False, True, True, False, False, True, True, True] # Actual tosses 23-32
sim_initial = False # Toss 22 result
sim_HT_prob = 0.6   # P(T|H)
sim_TH_prob = 0.5   # P(H|T)
print('Simulated Coin')
predict(sim_actual, sim_initial, sim_HT_prob, sim_TH_prob)
```