CMSC 389T HW2 SOLUTION

Phong Dinh and William Gasarch

Jan 5, 2017

Problem 4 Write a program in psuedocode that will scan a text ONCE and compute the vector of probabilities of letters.

There are multiple ways to solve this problem. You will earn full credits if you compute the probability vector in one-pass. You will lose credits if you just compute the frequency vector and not normalize that vector.

Algorithm 1 Compute probability vector in one pass

```
procedure COMPUTE_PROB_VECTOR(T)
    F ← array size 26
    for i = 1 to 26 do
        F(i) ← 0
    n ← length(T)
    for i = 1 to n do
        F(T(i)) ← F(T(i)) + 1
    for i = 1 to n do
        P(T(i)) ← F(T(i))/n
    return P
```

Problem 5 Write a program in psuedocode that will, given a text that has been coded by an AFFINE CIPHER outputs the original text (so in English). You will need to write a new version of IS-ENGLISH? and DECODE from the notes. You can assume that if a text is coded with AFFINE then $d(\vec{q}, \vec{p}) \sim 0.44$.

For this problem, you are supposed to use the 0.68 (from lecture notes) and 0.44 as we provided. If you don't use the these combinations, you lose 10 points.
You are supposed to decode affine text. You will lose 10 points if you don't do this.
We allow 0.02 tolerance.

Algorithm 2 Check if a long text is in English (using affine cipher for encoding / decoding)

```
procedure IS_ENGLISH_AFFINE(T)
    p⃗ ← prob vector of English
    q⃗ ← COMPUTE_PROB_VECTOR(T)
    DOT ← d(p⃗, q⃗)
    if DOT ≥ 0.66 then
        THIS IS ENGLISH
        return True
    else if DOT ≤ 0.42 then
        THIS IS NOT ENGLISH
        return False
    else
        THIS IS NOT ENCODED BY AFFINE CIPHER
        return False
```

Algorithm 3 Decode affine cipher given the ciphertext

```
procedure DECODE_AFFINE(T)
    n ← length(T)
    for a = 1 to 25 do
        if gcd(a, 26) = 1 then
            for b = 0 to 25 do
                for i = 1 to n do
                    T'(i) = a × T(i) + b (mod 26)
                if IS_ENGLISH_AFFINE(T') = true then
                    T' is the plaintext
                    return T' (and HALT the whole procedure)
```

Problem 6 STUDENT: Your method of decoding Shift Cipher is B*L*S*I*T! Just find the letter that occurs the most often and assume its e and go from there.

TEACHER: Okay, code that up and see how well it works.

So, help out this obnoxious student. Write pseudocode (like whats in english.pdf) that, given a text that IS a SHIFTED text finds the most freq letter and uses that to output the DECODED text.

Algorithm 4 Decode shift cipher with using the most frequent letter

$F \leftarrow$ array size 26
for $i = 1$ to 26 **do**
    $F(i) \leftarrow 0$
$n \leftarrow \text{length}(T)$
for $i = 1$ to n **do**
    $F(T(i)) \leftarrow F(T(i)) + 1$
$x \leftarrow$ The value of most frequent letter in T (We Omit how to find this from F but its easy.)
$s \leftarrow 4 - x \pmod{26}$ (NOTE: 4 is the value of 'e')
$T' \leftarrow$ SHIFT_TEXT(T, s)
return T'

By choosing the shift size as the difference between the most frequent letter in the cipher-text with 'e', we can decode the shifted text. If you find the most frequent letter and still make a loop to find the shift size, you lose all credits. The whole point of this question is avoid making that loop.

For your own benefit—you should complete this code by writing code to find the max of an array.

Problem 7 STUDENT: Your method of decoding Affine Cipher is B*L*S*I*T! Just find the two letters that occurs the most often, assume they are e and t and use those... somehow.

TEACHER: Okay, code that up and see how well it works.

So, help out this obnoxious student. Write pseudocode (like whats in english.pdf) that, given a text that IS an AFFINE-coded text finds the TWO most freq letters and uses them to output the DECODED text. You can assume that if a text is coded with AFFINE then $d(\vec{q}, \vec{p}) \sim 0.44$.

SOLUTION TO PROBLEM 7

We use that 'e' is 4 and 't' is 19. Lets say we know that e maps to y_1 and t maps to y_2. Then we have the following conditions on a, b:

Since we know $y_1 = ax_1 + b \pmod{26}$ and $y_2 = ax_2 + b \pmod{26}$, we have

$$\begin{cases} y_1 \equiv 4a + b \pmod{26} \\ y_2 \equiv 19a + b \pmod{26} \end{cases}$$

Thus, we have

$$y_2 - y_1 \equiv 15a \pmod{26}$$

We want to divide by 15. NO- we can't divide in mod 26. But we can MULTIPLY BY 15^{-1}. OH- does 15^{-1} EXIST? YES- by trial and error you can find out that its 7.

$$a \equiv (15)^{-1}(y_2 - y_1) \equiv 7(y_2 - y_1) \pmod{26}$$

From this we can find b:

$$\begin{aligned} y_1 &\equiv ax_1 + b \pmod{26} \\ b &\equiv y_1 - 4a \pmod{26} \\ b \equiv y_1 - 4 * 7 * (y_2 - y_1) \equiv y_1 - 2(y_2 - y_1) &= 3y_1 - 2y_2 \pmod{26} \end{aligned}$$

We use this in our algorithm.

In the algorithm below we omit the part where you FIND the two most frequent letters. This is similar to what we did for in the last problem to find the most frequent letter. (For your own benefit—you should complete this code by writing code to find the frequencies and then find the two most common ones.)

Algorithm 5 Decode affine cipher using two most frequent letters

procedure DECODE_SHIFT_TEXT(T)
$y_1 \leftarrow$ The value of most frequent letter in T
$y_2 \leftarrow$ The value of the second most frequent letter in T
$a \leftarrow 7(y_2 - y_1) \pmod{26}$
$b \leftarrow 3y_1 - 2y_2 \pmod{26}$
$T' \leftarrow$ AFFINE_TEXT(T, a, b)
return T'

Similar to the previous problem, you will lose all credits if you do not solve for a and b directly or still making two loops to try all possibilities.