Cryptography Assignment

Tess of the d'Urbervilles

Educational value

Or: other than a chance to score a really good coursework mark, what else does this get you?

This piece of coursework has two objectives:

- it gets you working in a practical sense with encryption, decryption, plaintext, ciphertext, and keys. This gives you a practical sense of what these are all about, which should be useful for the entirety of the crypto section of the module. It also lets you experience the impact of having certain quantities of ciphertext available.
- it makes sure you know some of the basic tricks in cryptology: substitution, permutation, possibly frequency analysis, applied to a few basic ciphers that you need to know about anyway.

All of this is essential in a course on cryptography, but much nicer to examine through coursework where you have time and computing resources available, rather than in an exam.
More importantly, this assignment is about problem solving. Can you, given a problem, come up with a solution to it by yourself. More than anything else, it is your ability to problem solve that will determine your success in your professional life and career.

Introduction

This assessment comprises seven exercises, unimaginatively called Exercise 1 to Exercise 7. For each exercise you will be given an extract from Thomas Hardy's novel *Tess of the d'Urbervilles* (first published in 1891), encrypted with one of the ciphers we discussed earlier in the course. Your job is to decrypt the extract, recovering the plaintext. In each exercise, every student will be given a different extract to decipher, and the encryption key will vary from student to student; however, all the extracts within a particular exercise will be encoded with the same sort of cipher, as described below. For example, in Exercise 1 everyone will have an extract encoded with a Caesar cipher.

You are free to work with colleagues in devising methods for decrypting particular sorts of cipher, and in putting together programs to help you do so; however, **when it comes to decrypting your own individual pieces of ciphertext, you must work individually, without help from anyone else**.

Brute-force methods are not ruled out; however, you may find the assessment more rewarding if you introduce an analytical component into your decryption tactics.

Preliminaries

On `raptor`, in the directory `\\raptor.kent.ac.uk\exports\courses\comp5580\` (or equivalent `/courses/comp5580/`) you will find the following files:

`tess.txt`
: This is the ASCII text of *Tess of the d'Urbervilles* from the Project Gutenberg website. Provided mainly for interest, and for the licensing information at the end.

`tess27.txt`
: This is `tess.txt` reduced to a 27-character alphabet in the following way:

1. Prefatory and ending material from Project Gutenberg that isn't part of the novel itself has been removed;
2. Apostrophes have been removed;
3. Lower-case letters have been converted to upper case;
4. Each sequence of one or more non-letter characters (including for example digits, punctuation, hyphens and whitespace) has been replaced by a single '|' (vertical bar) character.

`tess26.txt`
: A further reduction to a 26-character alphabet, obtained by omitting the vertical bars from `tess27.txt`.

For most exercises, the plaintext is a randomly chosen string of 840 characters taken from `tess26.txt`; the exceptions are Exercise 2, where the extract is only 30 characters long, and Exercise 7, where the extract comes from `tess27.txt`. **You therefore cannot assume that the extract will start at the beginning of a word in the original novel, nor that it ends at the end of a word.**

The Exercises

In the `assignment` sub-directory, you will find further subdirectories named with your login. Within each of these directories you should find seven text files in the form `cexercise1.txt` etc. (If you don't, contact c.perez@kent.ac.uk as soon as possible.) These files contain the ciphertexts that you have to decrypt for the exercises in question, terminated by a newline which does not form part of the ciphertext.

Exercise 1 (2 marks)
: The plaintext comes from `tess26.txt` and is encoded with a Caesar cipher.

Exercise 2 (3 marks)
: The plaintext comes from `tess26.txt` and is encoded with a Vigenere cipher using the 21-letter key `TESSOFTHEDURBERVILLES`.

Exercise 3 (4 marks)
: The plaintext comes from `tess26.txt` and is encoded with a Vigenere cipher. The key is an arbitrary sequence of six letters (i.e. not necessarily forming an English word).

Exercise 4 (5 marks)
: The plaintext comes from `tess26.txt` and is encoded with a Vigenere cipher. The key is an arbitrary sequence of between 4 and 6 letters.

Exercise 5 (5 marks)

The plaintext comes from `tess26.txt` and is encoded with a transposition cipher, as follows: the plaintext is written row-wise across a certain number of columns, between 4 and 6. (You must figure out how many columns were used.) The ciphertext is formed by reading out successive columns from left to right.

Exercise 6 (5 marks)

The plaintext comes from `tess26.txt` and is encoded with a transposition cipher, as follows: the plaintext is written row-wise across six columns. The ciphertext is formed by reading out successive columns in an arbitrary order (which you must figure out to decipher the message). *Hint:* look for common pairs of letters, such as 'th'.

Exercise 7 (6 marks)

The plaintext comes from `tess27.txt` and is encoded with a general substitution cipher, using a randomly chosen mapping from the 27-character alphabet onto itself. Note that normally (i.e. except by chance) a vertical bar will be mapped onto some other letter of the alphabet.

Not Cricket

The files `tess26.txt` and `tess27.txt` are made available to you to enable you to *check* your answers, not to assist you in finding the answers. For example it would be contrary to the spirit of this assessment to write programs to search these files as part of the decryption process. It is not straightforward to detect this, but if there is evidence, marks will be deducted.

However, it is entirely in order to analyse these files to determine language statistics, e.g. letter frequencies.

Use of Software

You are not expected to do any decryption by hand. You are strongly encouraged to write your own software (individually or in groups). You may use ANY programming language you desire for this task, from C to Python and Java, and even very high-level languages like Matlab or Mathematica.

If you prefer not to do any programming, you may also use spreadsheet software like MS Excel. ***The use of any other software is strictly prohibited!***

Regardless of what you decide to use, you do NOT need to submit any of the software you used. See more below, in the submission section.

Dyslexia

As mentioned above, you are NOT expected to solve the problems by hand. You should be automating your solution as much as possible. Writing a program, and then using copy-paste to input the cyphertext to your program (or Excel sheet, etc) should sidestep any issues due to dyslexia.

That said, it is possible that dyslexia may make one or two of the exercises particularly difficult. If, having made a serious attempt, you find that you cannot make progress with an exercise because of this, please contact me (c.perez@kent.ac.uk).

How to Submit Your Work

The deadline is on the Moodle page as well as Kent Vision. Deadline on the Moodle takes precedence over any other information). In `/proj/comp5580/decryption/` (or `\\raptor.kent.ac.uk\exports\proj\comp5580\decryption\`), you will find a folder with your login on it. Into this folder you should place a text file called `exercise1.txt` with your answer to Exercise 1. (To reiterate, it's got to be a *text* file and be called `exercise1.txt`: calling it `Exercise1.txt`, `excercise1.txt`, `exercise1.doc` or `exercise1.txt.txt` ("hide known extensions" - you're doing a module on Security!?) etc. is a surefire way to get 0 marks for Exercise 1.) Similarly place in this folder files called `exercise2.txt`, ... `exercise7.txt`.

Each of these files must have the following format:

- Line 1 must contain **(only) the first 30 characters** of the plaintext you have obtained by decrypting your ciphertext extract. To get full marks for an exercise, it is necessary that these characters be exactly right, so it's worth checking that they correspond to a sequence of 30 characters from `tess26.txt` or `tess27.txt` as the case may be. **Indeed it will normally be necessary to get these characters exactly right to get *any* marks from the exercise.**
- Line 2 must be blank.
- Lines 3 **onwards** should contain a description of how you carried out the decryption. This description need not be very long - in some cases a single sentence may do - but should be sufficient to allow a reader to reproduce your approach. If you used a third-party program to help you, you should identify it. And you need to be clear on how you used the program, and what the program was doing for you (*e.g. "I used MS Excel to…").* If you wrote your own program, you should provide an English readable description of the algorithm (*e.g. "My program calculates the relative frequency of each letter in the tess26.txt file in the following way…").*

  It will not be possible to look at these descriptions in every case, but spot checks will be made, and if the description/source is missing or inadequate, marks shall be deducted. I some rare cases I may have to ask you to submit your code separately for review, in which case failure to do so may also result in forfeiture of marks.

For example, this is what `exercise1.txt` might look like:

```
RINTANDFROMTHEBACKOFHERHEADAKI

Using pen and paper, I worked through all possible shifts until
I found one that converted the ciphertext into recognisable
English. I checked that the resulting extract occurred in
tess26.txt. All this took some time, so I then boarded the
Tardis and travelled back to the beginning of the 21st century
to submit my work.
```

If accessing Raptor remotely (or even from some subnets on campus) you will need to be running the University provided VPN software on your computer! Make sure your VPN

software is installed. *No submissions will be accepted through other means than by uploading to Raptor. It is your responsibility to ensure you are able to connect to Raptor.*

Checking your submission using Kentsubmit

This is assignment uses the Kentsubmit system. This system was created in order to help you submit your coursework properly, help you verify your submission meets all the submission criteria, correct any possible errors, and give you the peace-of-mind when all is said and done of knowing that your assignment is properly submitted.

To run it, open a command-line terminal connection to raptor.kent.ac.uk, and run the command "`kentsubmit`" (no quotation marks), and then follow the on-screen instructions to choose your module (comp5580) and the assignment (decryption), kentsubmit will then check your assignment is properly submitted.

Technically, you do not need to use `kentsubmit`–a properly submitted assignment will be marked properly regardless of whether or not you used `kentsubmit`. However, using `kentsubmit` will ensure you've submitted correctly. Please note that you can modify your submission / resubmit any number of times before the submission deadline. Once the deadline has passed, you will have to request an extension from the student office, and submit into the late directory (see next section).

Please note: that if you fail to use kentsubmit, or use it and ignore its warnings, and submit your files wrong (wrong location, name, format, etc) you are risking getting a zero on those questions (possibly the whole assessment). Given the tools provided to you to ensure proper submission, there will be no exceptions to the marking scheme made due to improper submissions. That said, if kentsubmit were to somehow miss an issue with your submission, and you lose marks due to submission formatting that kentsubmit failed to pick up (as unlikely as that is), I will personally fix your submission, and award you full marks for that question.

Late Submissions

In order to obtain a deadline extension for the assignment, you must contact the student office directly at:

https://moodle.kent.ac.uk/2022/course/view.php?id=94

You can submit your late assignment into the late submission directory:

```
proj/comp5580/decryptionlate/
```

This directory will remain open until precisely one week after the original submission deadline (since no extensions can go beyond one week). All late submissions will be marked, and the mark passed on to the student office. However, the student office will only recognise the late marks of those who have been granted an extension.