**《信息安全概论》实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** | **谢双骏** | | **年级** | | **2019级** |
| **学号** | **20194196** | | **专业、班级** | | **计算机科学与技术01班** |
| **实验名称** | **频度分析法的实现** | | | | |
| **实验时间** | **2022.10.6** | **实验地点** | | **线上** | |
| **实验成绩** |  | **实验性质** | | **□验证性 □设计性 □综合性** | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  评语：  评价教师签名（电子签名）： | | | | | |
| **一、实验目的**  掌握频度分析法原理和RSA加解密原理 | | | | | |
| **二、实验项目内容**  1.使用频度分析法解密以下文本，并给出替换表：  WTWHEQTOKEBPELHTPUKOKEKCKCWIQTKCTPTOKREKQTTQHFEKJQWMWMRUKBPEKLHTOQTBEPJTOKHKOPMPEKCCKQCZKTQFKWMIEKQHKCCKAPTWPMTPTOQTIQLHKBPEZOWIOTOKDRQAKTOKXQHTBLXXJKQHLEKPBCKAPTWPMTOQTZKOKEKOWROXDEKHPXAKTOQTTOKHKCKQCHOQXXMPTOQAKCWKCWMAQWMTOQTTOWHMQTWPMLMCKERPCHOQXXOQAKQMKZUWETOPBBEKKCPJQMCTOQTRPAKEMJKMTPBTOKSKPSXKUDTOKSKPSXKBPETOKSKPSXKHOQXXMPTSKEWHOBEPJTOKKQETO | | | | | |
| **三、实验设计**  **1. 频度分析原理**  对于任何一种书面语言而言，不同的字母或字母组合出现的频率各不相同。如果以这种语言书写足够长的文本，都呈现出大致相同的特征字母分布规律，如下图所示：  表格  描述已自动生成  在上表中，不少字母出现的概率近乎相等。为了应用方便，常将英文字母表按字母出现的概率大小分类，分类情况见下表：  --------------------------  极高频 E  次高频 T A O I N S H R  中等频 D L  低频 C U M W F G Y P B  甚低频 V K J X Q Z  --------------------------  语言的单字母统计特性没有反映出英文双字母和多字母的特征，在双字母中统计出概率最大的30对字母按概率大小排列为：  th he in er an re ed on es st en at to nt ha nd ou ea ng as or ti is et it ar te se hi of  类似的，我们还可以考察英文课文中三字母出现的频率。按Beker在1982年统计的结果(样本总数100 360)得到概率最大的20组三字母按概率大小排列为：  the ing and her ere ent tha nth was eth for dth hat she ionhis sth ers ver  特别，the出现的频率几乎为ing的3倍。 | | | | | |
| 1. **实验过程或算法** 2. **频度分析法解密** 3. **去除多余字符**   *#去除英文之外的其他字符 def* remove\_other\_characters(sentence):  *return* re.sub(u"([^\u0041*-*\u005a\u0061*-*\u007a])", "", sentence)   1. **统计所给字符串中的词频**   *# 计算字符频率 def* calculate\_frequency(sentence):  temp\_list = {}  temp\_list2 = {}  temp\_list3 = {}  count = 0  *for* i *in range*(0, *len*(sentence)):  count += 1  t1 = sentence[i]  *if* i < *len*(sentence) - 1:  t2 = t1 + sentence[i + 1]  *if* i < *len*(sentence) - 2:  t3 = t2 + sentence[i + 2]  *if* t1 *in* temp\_list:  temp\_list[t1] += 1  *else*:  temp\_list[t1] = 1  *if* t2 *in* temp\_list2:  temp\_list2[t2] += 1  *else*:  temp\_list2[t2] = 1  *if* t3 *in* temp\_list3:  temp\_list3[t3] += 1  *else*:  temp\_list3[t3] = 1  *# for key in temp\_list.keys():  # temp\_list[key] = temp\_list[key] / count  # print(temp\_list)* temp\_list = *sorted*(temp\_list.items(), key=*lambda* kv: (kv[1], kv[0]), reverse=*True*)  temp\_list2 = *sorted*(temp\_list2.items(), key=*lambda* kv: (kv[1], kv[0]), reverse=*True*)  temp\_list3 = *sorted*(temp\_list3.items(), key=*lambda* kv: (kv[1], kv[0]), reverse=*True*)  *# print(temp\_list)  print*("各个字符出现的次数:")  *for* element *in* temp\_list:  *print*(element[0] + ":" + *str*(*float*(element[1] / count)) + "次")  segmentation()  *for* element *in* temp\_list2:  *print*(element[0] + ":" + *str*(element[1]) + "次")  segmentation()  *for* element *in* temp\_list3:  *print*(element[0] + ":" + *str*(element[1]) + "次")  segmentation()  得到结果：  各个字符出现的次数:  K:56次  T:40次  O:32次  Q:29次  P:28次  E:22次  W:17次  M:17次  C:17次  H:16次  X:14次  B:11次  A:8次  S:7次  R:6次  L:6次  J:6次  Z:4次  U:4次  I:4次  D:3次  F:2次  --------------------------------------------------------------TO:21次  OK:12次  OQ:11次  QT:10次  EK:10次  ......  --------------------------------------------------------------  TOK:10次  TOQ:7次  OQT:6次  ETO:5次  BPE:4次   1. **替换字符串中的K和T**   **QQ截图20221018221015**  （replace函数为自己实现的字符串替换函数，代码如下图）  LAY~1[EQ9YMB3P9C9HM59MH  替换结果如下图：  QQ截图20221018221202   1. **由于“WT”替换为“?T”且?T处于开头，容易推出该单词为IT，所以推出W->I。并替换原字符串中的W。**   **QQ截图20221018221535**   1. **由于“WTWHEQTOKE”替换为“IT I? RATHER”，容易推出该单词为IS，所以H->S。并替换原字符串中的H。**   **QQ截图20221018221626**   1. **由于BPE频率很高，且E->R，因此猜测BPE->FOR,B->F,P->O。**   **QQ截图20221018221829**   1. **由于“UKBPEK”为“ ?EFORE?”，猜测该单词为BEFORE，所以U->B。**   **QQ截图20221018222037**   1. **由于“LH”为“ ?S”且?S夹在中间，因此L->U**   **QQ截图20221018222222**   1. **由于“IQLHK”为“ ?AUSE”，猜测该单词为CAUSE，所以I->C**   **QQ截图20221018222354**   1. **由于解出的单词中出现了“FREED?M”，因此暂时认为C->D。**   **QQ截图20221018222454**   1. **由于“QMC”为“A?D”，推测M->N。**   **QQ截图20221018222558**   1. **由于“CKAPTWPM”变为“DE?OTION”猜测为DEVOTION，即A->V**   **QQ截图20221018222709**   1. **由于“RPAKEMJKMT”变为“?OVERNMENT”猜测为GOVERNMENT，即R->G**   **QQ截图20221018222838**   1. **由于“SKEWHO”变为“?ERISH”猜测为PERISH，即S->P**   **QQ截图20221018222950**   1. **由于“SKPSXK”变为“PEOP?E”猜测为PEOPLE，即X->L**   **QQ截图20221018223110**   1. **由于“OWROXD”变为“HIGHL?”猜测为HIGHLY，即D->Y**   **QQ截图20221018223212**   1. **由于“TQHF”变为“TAS?”猜测为TASK，即F->K**   **QQ截图20221018223313**   1. **由于“ZOWIOD”变为“?HICH”猜测为WHICH，即Z->W**   **QQ截图20221018223425**  **至此字符串已解密完成。** | | | | | |
| 1. **实验过程中遇到的问题及解决情况**   本次实验比较简单，问题不是很多：   1. 频度分析法一些单词很生僻，比如DEDICATED，无法通过经验尝试出来；遇到这种情况，不要死扣这一个单词，因为基本上他的真实值不在你的词汇中，应该及时更换目标，结合词频和语法去分析其他单词的替换。 | | | | | |
| 1. **实验结果及分析和（或）源程序调试过程** 2. 频度分析法解密 3. 解密结果   IT IS RATHER FOR US TO BE HERE DEDICATED TO THE GREAT TASK REMAINING BEFORE US THAT FROM THESE HONORED DEAD WE TAKE INCREASED DEVOTION TO THAT CAUSE FOR WHICH THEY GAVE THE LAST FULL MEASURE OF DEVOTION THAT WE HERE HIGHLY RESOLVE THAT THESE DEAD SHALL NOT HAVE DIED IN VAIN THAT THIS NATION UNDER GOD SHALL HAVE A NEW BIRTH OF FREEDOM AND THAT GOVERNMENT OF THE PEOPLE BY THE PEOPLE FOR THE PEOPLE SHALL NOT PERISH FROM THE EARTH   1. 替换表  |  |  | | --- | --- | | 密文字母 | 明文字母 | | A | V | | B | F | | C | D | | D | Y | | E | R | | F | K | | G | / | | H | S | | I | C | | J | M | | K | E | | L | U | | M | N | | N | / | | O | H | | P | O | | Q | A | | R | G | | S | P | | T | T | | U | B | | V | / | | W | I | | X | L | | Y | / | | Z | W |   “\”表示字符串中没有出现的字母 | | | | | |