Univerza *v Ljubljani*
Fakulteta *za računalništvo*
in informatiko

Cryptography and computer safety

Tools to decipher the Elizika letter

Author:
Erik Janežič

Ljubljana, August 23^{rd} 2017

Contents

Summary

This project focuses on developing tools to ease manual decoding of substitution cyphers and explores ways to automate the decoding process. Simultaneously we try to decipher an old encoded letter named *Elizika* from the early 20th century. We present various text analysis tools to help with decoding and an algorithm for automatic breaking of substitution cyphers. For now this algorithm works only with English language cyphers due to lack of good corpora for Slovenian language. At the end some future plans and ideas for improvement are presented.

Povzetek

Fokus tega projekta je razvoj orodij za olajšanje ročnega dekodiranja substitucijskih šifer ter poskus avtomatiziranja celotnega procesa. Hkrati skušamo razvozljati šifro starega pisma imenovanega *Elizika* iz začetka dvajsetaga stoletja. Predstavili bomo različna analitična orodja, ki nam bodo olajšala dekodiranje in algoritem za avtomatsko razbijanje substitucijskih šifer. Za enkrat algoritem deluje le za kodirane texte v Angleškem jeziku, ker za Slovenski jezik primanjkuje primernih korpusov. Na koncu predstavimo nekaj planov in idej za izboljšanje obstoječih algoritmov.

1 Preface

The motivation for this work lies in our curiosity for unraveling old secrets. We chose to tackle the challenge of deciphering an old encrypted letter from the early 20th century. Our goal was to develop tools that will simplify or potentially even automate the decryption process of substitution cyphers. Tools such as Black Chamber by Simon Singh already exist on the web but they are usually bulky to use, lack responsive visualization, use unsatisfactory cypher text analysis (some automated tools just take frequency analysis into account)and are limited to small group of popular languages. This project focuses on developing the prototype tools in the Python language and testing them on custom generated cypher texts, which will ideally be transfered to a web application at later time.
This paper is structured in the following way; Initially we will focus on the meta information (origin, language, writing stile,...) extractable from the photography of the original encrypted letter. In the next step we describe the process for determining the cypher type, followed by different cypher text analysis and description of decryption tools:

- N-tuple analysis
- Text probability analysis
- Vowel analysis
- Dictionary attack
- Trial and error based on gathered data
- Fully automated approach

At the end we will introduce more detailed future plans for the project.

2 Background

First step in every decryption task is to examine the physical traits of the original cypher text (if we have one, else we can skip directly to cypher text analysis).
Examining the *Elizika letter* we see that in addition to the cyphered content of the letter we have clearly visible recipient information, a stamp, writers greeting, farewell and his signature. First issue of the stamp used in a letter was in the year 1908 and they stopped producing it in 1916 so we can assume that the letter was written in this period.From the stamp seal we can see that the letter was sent from a city Laibach (present Ljubljana), which was at that time a part of Austrian-Hungarian empire. The recipients name was Elizika Tihelj whose residence address was Bled no. 7 in the Gorenjska region (northern part of nowadays Slovenija). Writers style of addressing the recipient *"Velecenjeni gospodični"* (meaning *"highly esteemed"*) indicates his admiration towards her. His admiration is further implied in his greetings *"Draga mi!"* (meaning *"My dear"*. Author concludes the letter with a phrase *"Na svidenje"* which literally translates to *"Till we see each other again"* and is a common salutation phrase in Slovenian language. From this meta data we can assume that the letter is written in early 20th century Slovenian language and that the contents is most likely personal in nature.

Figure 1: Colorized stamp from the letter

Figure 2: Information about the recipient and authors greeting on the left

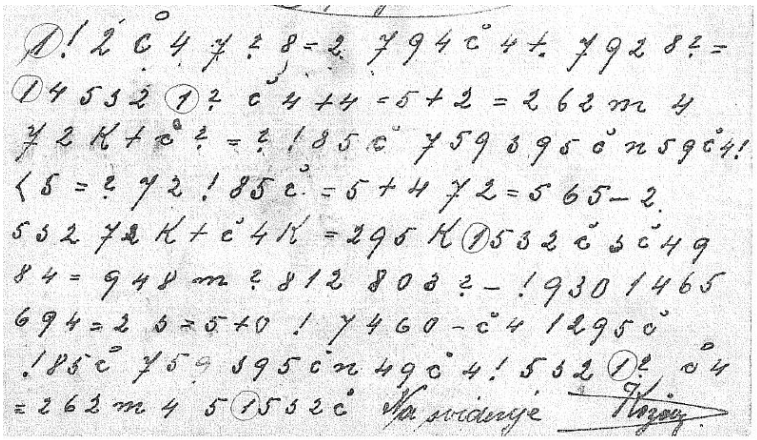


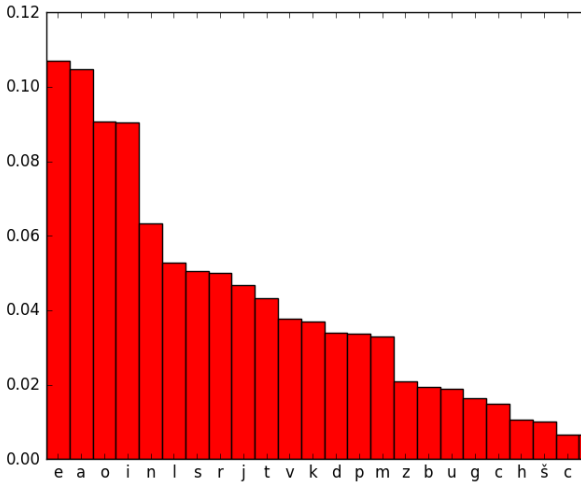
Figure 3: Cyphered content of the letter with authors salutation and signature at the end

3 Determining the cypher

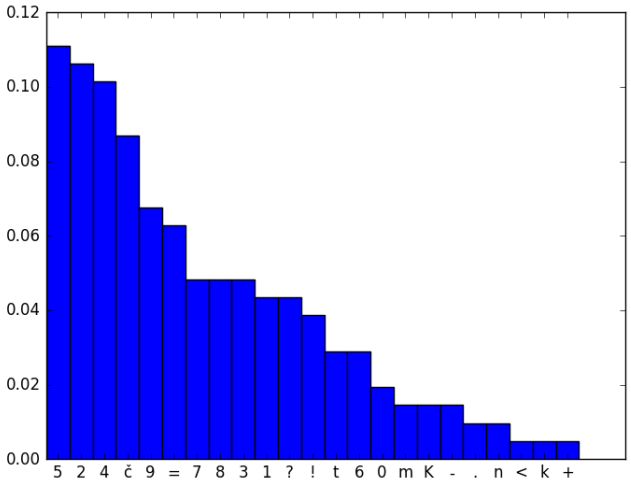
First goal of each deciphering problem is to determine the cypher type. Natural language attributes such as letter frequencies and index of coincidence can quickly indicate whether we are likely to deal with a substitution cypher. Since we assume the letter was written in Slovenian language all natural language text characteristics were compared to characteristics of a Slovenian text. In the first step we digitalise the cypher text, so it can be used in the analysis. This yields the following interpretation interpretation:

1!2č47?8=2.794č4t.7928?=
145321?č4t4=5t2=262m4
72ktč?=?!85č759395čn59č4!
<5=?72!85č=5t472=565-2
53272ktč4k=295k1532č3č49
84=948m?812808?-!9301465
694=23=5+0!7460-č41295č
!85č759395čn49č4!5321?č4
=262m451532č

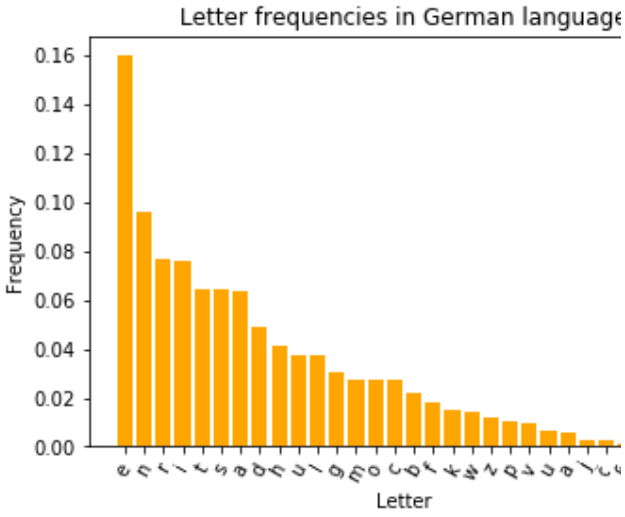
Calculating letter frequencies is done straightforwardly by counting number of occurrences for each letter and dividing it by the number of all letters in the text. In image 4 we visualized character frequencies of the cypher text, Slovenian, Hungarian, German and Italian language. By analyzing the shape of the graphs we can probably exclude German and Hungarian language as potential languages. Graph shapes for cypher text, Slovenian and Italian language are more similar. Closer inspection reveals to us that in contrast to a frequency drop at 5th character for Slovenian and cypher text frequencies there is a frequency drop at 4th character for Italian frequencies. Furthermore we can see that Slovenian and cypher text frequencies decrease with a slower rate after the 5th character than the Italian frequencies. Since the resemblance between Slovenian frequencies and cypher text frequencies is the strongest we can assume that we are dealing with a substitution cypher and a Slovenian content. From the letter frequency analysis we can also suspect that there are no character for space in the cypher text since expected space frequency is around 17%.



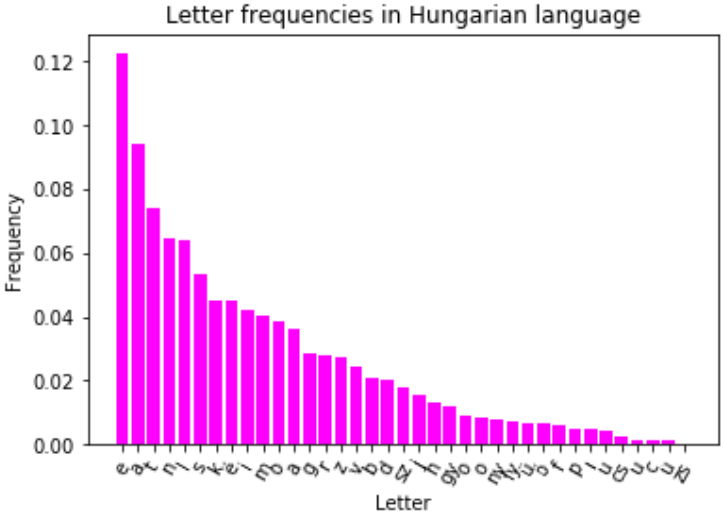
(a) Frequency of Slovenian letters



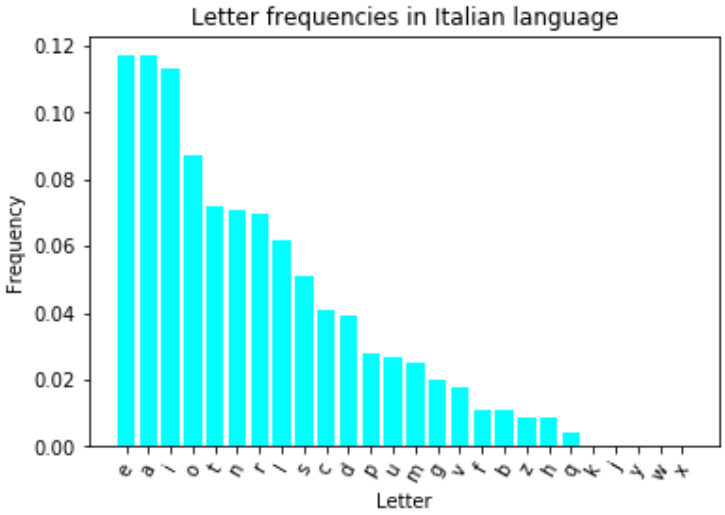
(b) Frequency of cyphered letters



(c) Frequency German letters



(d) Frequency of Hungarian letters



(e) Frequency of Italian letters

Figure 4: Comparison of cypher text letter frequencies and Slovenian letter frequencies

To reinforce this finding index of coincidence (IC) was calculated for the cypher text. Index of coincidence is calculated by the following formula:

$$c \times \sum_{i=1}^{c} \left(\frac{n_i}{N} \times \frac{(n_i - 1)}{(N - 1)} \right)$$

Where $"c"$ is the number of different letters in the alphabet $"N"$ is the length of the text and $"n_i$ is the number of times each letter appears in the text. Cypher text IC was compared against IC of a collection of Ivan Cankars literary works in two different forms; separated words and joined words.

Comparison of indexes of coincidence		
Cankar separated	Cankar joined	Cypher text
1.96508	1.65314	1.65093

As seen in table 1 there is a high IC similarity between cypher text and Cankar joined text, indicating again that we are most likely dealing with a substitution cypher.

4 N-tuple analysis

In each natural language some words are more common than others. Same holds for pairs/triplets of subsequent letters. Due to the fact that our cypher text is relatively short and N-tuple frequencies start to differ drastically less and less when $N > 3$ we conclude that only $2-tuples$ and $3-tuples$ give us useful additional information.
Since we suspect that our cypher text has no spaces, I compared pair and triplet frequencies against joined Cankar texts and seperated Cankar texts again. The results are seen in the following figures.

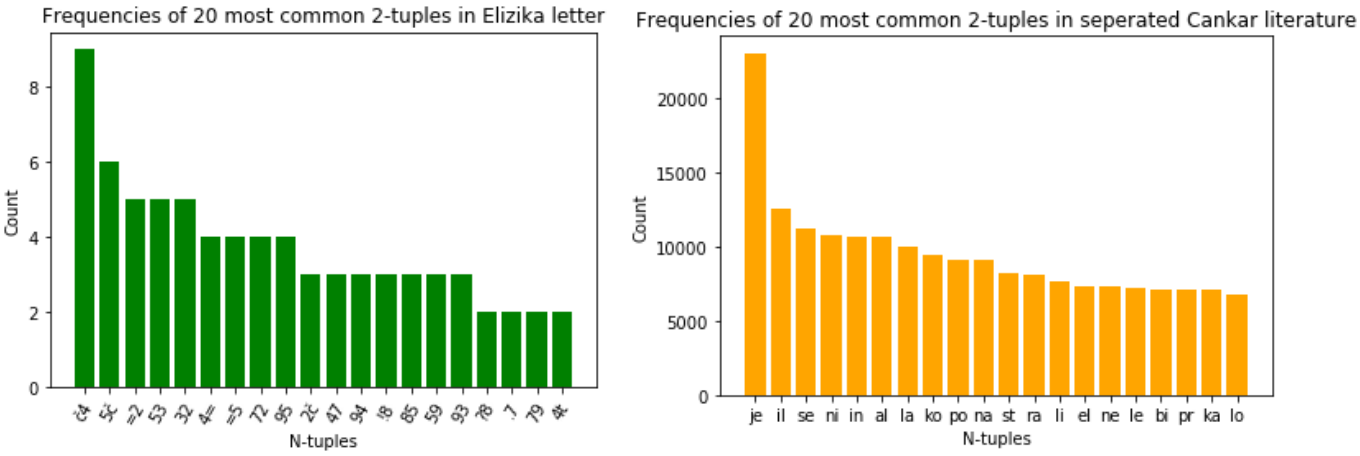


Figure 5: Comparison of cypher text 2-tuples with Cankar separated text 2-tuples

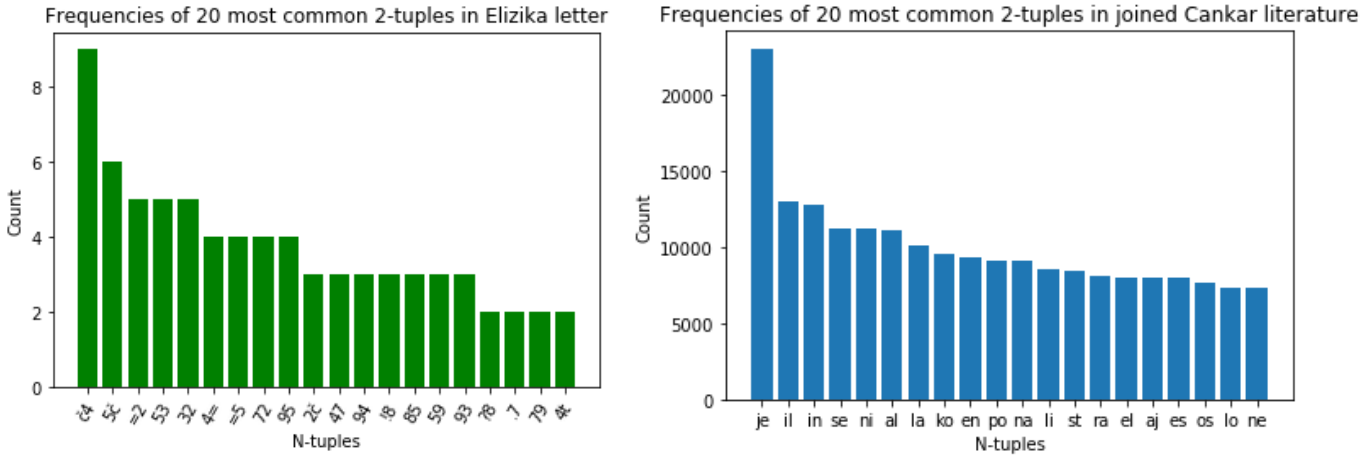


Figure 6: Comparison of cypher text 2-tuples with Cankar joined text 2-tuples

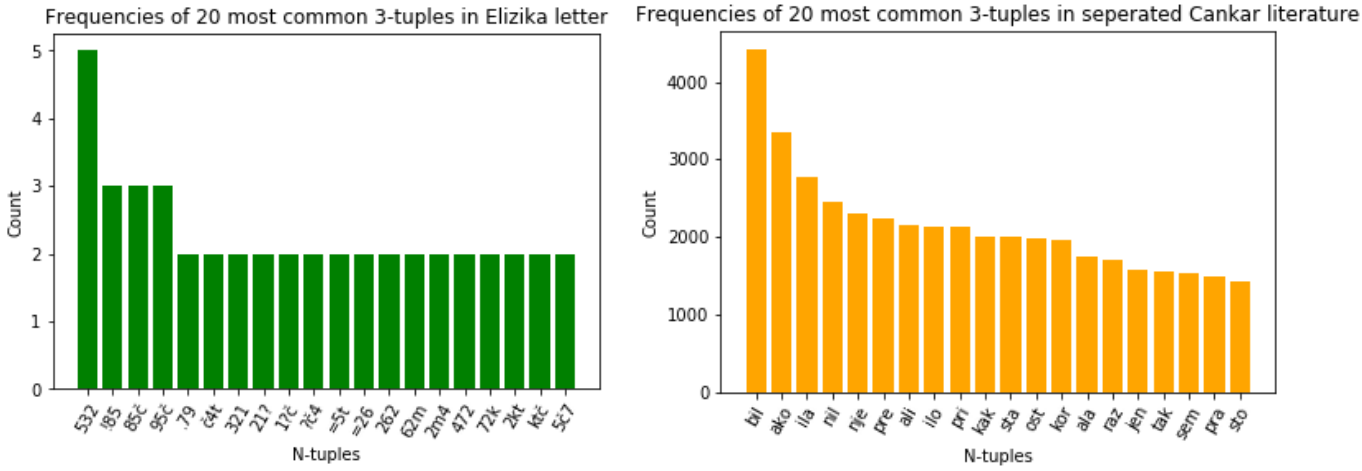


Figure 7: Comparison of cypher text 3-tuples with Cankar separated text 3-tuples

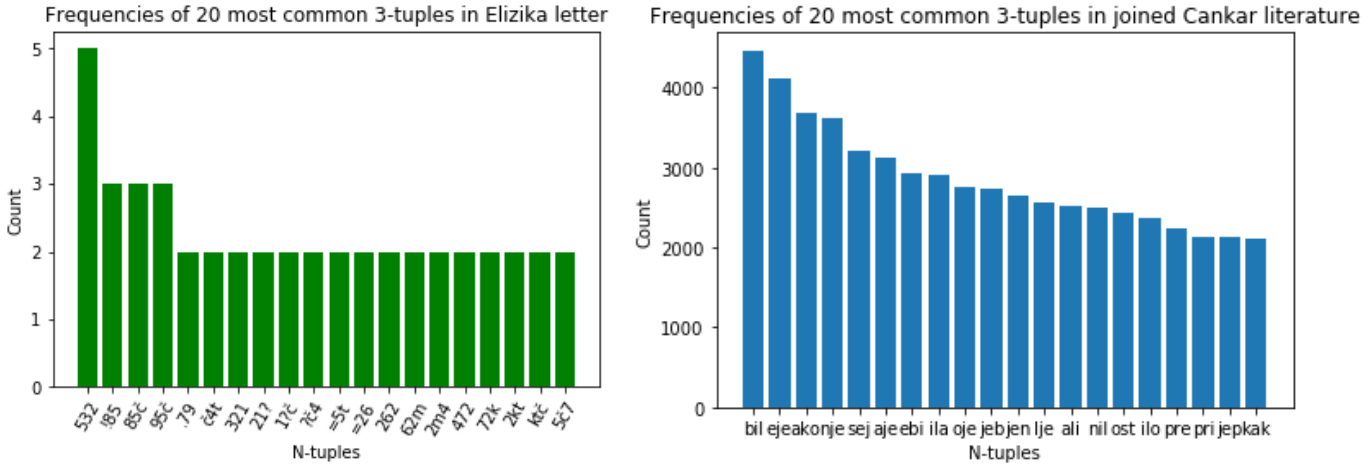


Figure 8: Comparison of cypher text 3-tuples with Cankars joined text 3-tuples

Produced charts will give us a better idea when manually determining character mappings. We must take into account that Cankar was mostly writing in the past tense, thus making words such as *"bil"* probably much more common than in our cypher text, which might be written in present or future tense.

For illustration we show on figures 4 and 5 that taking N-tuple frequencies, where $N > 3$, as a guide for choosing potential character mappings is pointless, due to small amount and unreliable additional information.

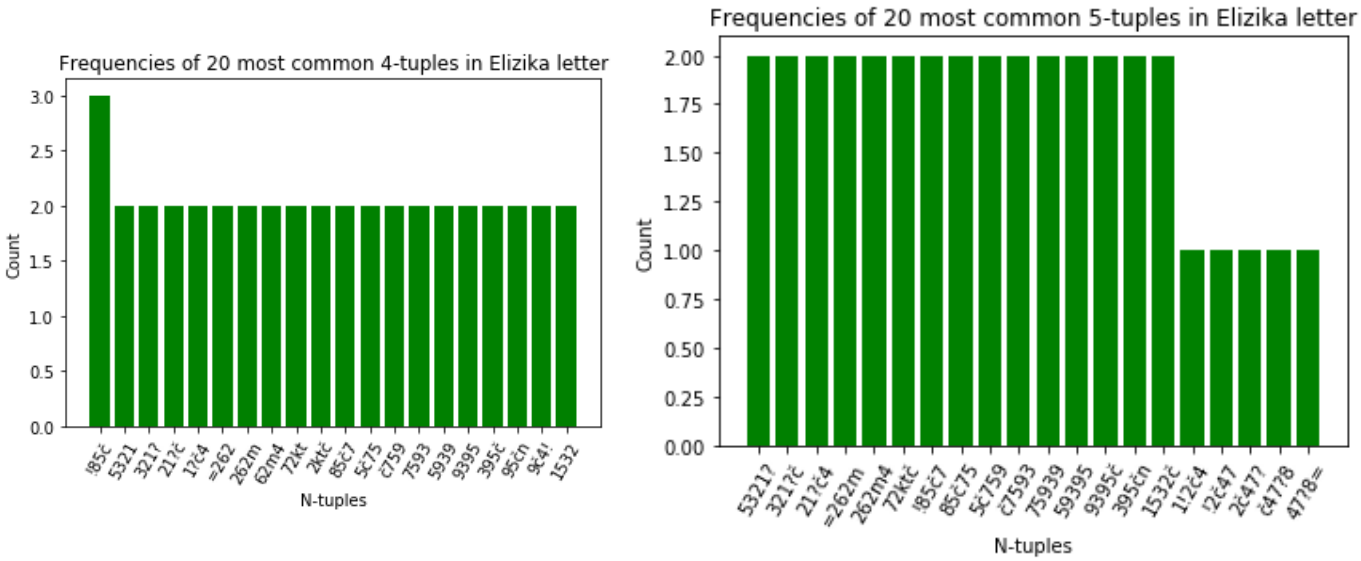


Figure 9: Frequency differences among 4 and 5-tuples in cypher text differ to little to give us additional information about character mappings

Next step was to search for all N-tuples in cypher text that occur at least 2 times and graphically marking them on the original cypher text. This gives us the idea which sequences of letters represent individual words or phrases. The result is seen on figure 10.

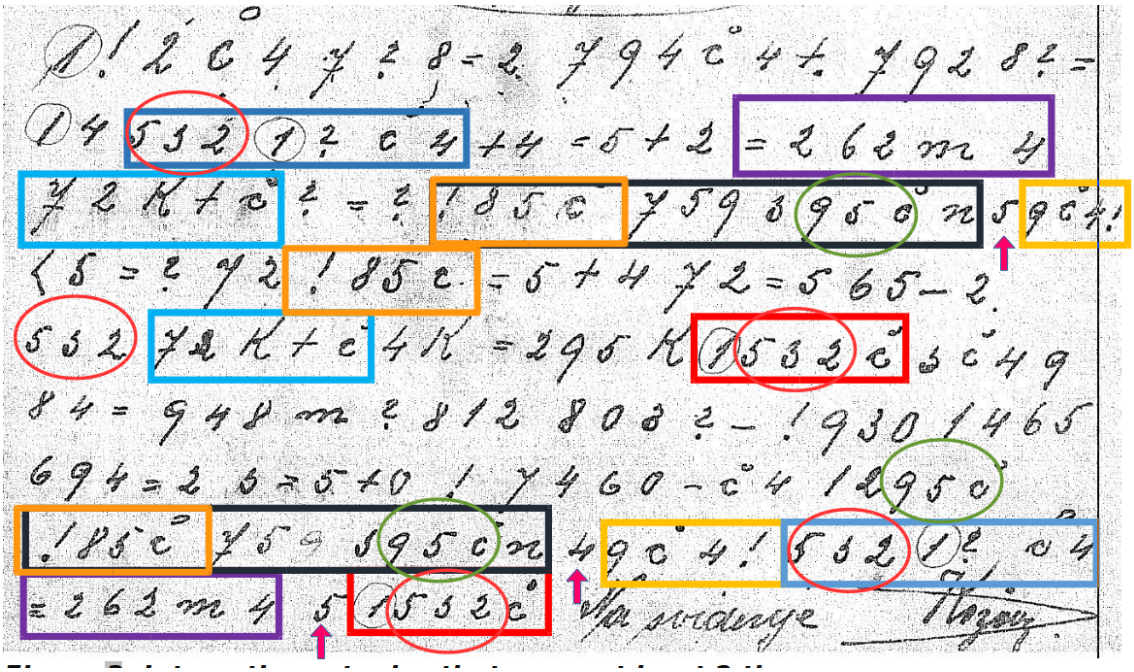


Figure 10: Graphic overlay of N-tuples that occur at least 2 times

We can observe quite some interesting features in figure 10. In addition to seeing where words might start and end we also see one long sequence, namely *"!85č759395čn"* which has parts of it occurring outside the sequence itself as well, indicating that it is probably

a multi word phrase. Another interesting feature are the single letters between two words marked with red arrows. These most likely represent single letter affixes indicating different sex, tense or quantity. In Slovenian language a,i and e are most common affixes expressing for such traits, thus giving us evidence for potential maps.

5 Text probability analysis

In this step we try to automate the discovery of promising character maps. Natural language text have this property that some letters are more likely to follow certain letters than the others. To exploit this trait in favor of evaluating decoding maps we construct a collection of dictionaries that contain the probabilities of one letter following the other based on the words in Slovenian dictionary. Whit this collection of dictionaries we are then able to evaluate the likely hood of some text being written in a natural language. For reference text I have chosen the joined Cankar collection again since our text probably doesn't contain spaces as well.
The probability of text was evaluated based by the following formula:

$$\frac{\sum_{i=1}^{N}(P(n_i, n_{i+1})}{N}$$

Where N is the length of the text an $P(n_i, n_{i+1})$ is the probability that a letter n_{i+1} follows a letter n_i.
Calculating text probability of reference texts gives us probability values of around 0.105 while poorly deciphered texts give us values in the range from 0.06 to 0.075. Using this heuristic for evaluating decoding maps we construct a function that slowly builds towards better and better maps with a brute force like approach. After letting the code run over the night, I was left with some maps with relatively high text probability of which the mapping seen in table 2 had the highest value of 0.1006225.

Table 1: Map with highest probability heuristic

+	6	7	n	<	3	5	k	9	m	0	č	4	=	.	?	k	2	t	1	-	!	8
f	d	j	c	u	r	e	ž	p	g	s	n	o	l	š	i	h	a	m	v	č	k	t

This map produced the following deciphered text:

vkanojitlašjponomšjpatil
voeravinomolemaladago
jažmniliktenjeprpencepnok
uelijaktenlemojaledeča
erajažmnožlapežveranrnop
tolpotgitvatstičkprsvode
dpolarlefskjodsčnovapen
ktenjeprpencopnokeravino
ladagoeveran

Although the decryption is clearly wrong, the tool for finding promising maps might still prove useful during manual deciphering.

6 Vowel analysis

Another important trait of natural texts is distribution of spacings between two subsequent vowels. Determining this distribution gives us information about which characters are more likely to be vowels.
For computing vowel distance distributions we use Cankar joined texts again. We count how many times vowel gaps of different lengths appear in the text and divide it by the total number of gaps to obtain vowel gap distributions seen in figure 11.

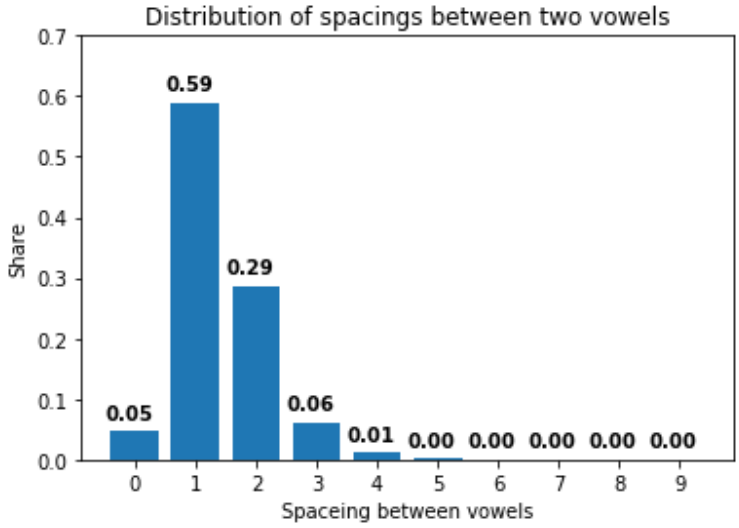


Figure 11: Vowel gap distributions in Cankar joined text

When reference distribution was constructed we produce every possible combination of decryption maps (containing only vowels, other characters were not important for this analysis and marked with "*") and calculated the vowel distribution for deciphered text obtained by using each one of those maps. We then calculate the similarities between decrypted text vowel distributions and reference vowel distribution by the following procedure:

- For each decrypted text we calculate the sum of square differences compared to the reference gap distributions of Cankar joined text.
- If gap of length N from the reference distribution was not present we add 0 to the total sum
- If gap of length N from decrypted text was not present in the reference distribution the square distance was penalized more by multiplying it by 100. This way we ensure that long vowel gaps don't occur since the probability of gap of length 9 for example in natural text is only $6.03 * 10^{-6}$.

The decryption maps are then order based on the lover sum of square differences. Values of sums of squared differences at the top of the list didn't differ drastically. For illustration you can see the most promising map and the corresponding distribution below (figure 12).

Table 2: Most probable vowel candidates

+	6	7	n	<	3	5	k	9	m	0	č	4	=	.	?	k	2	t	1	-	!	8
*	*	*	*	*	*	o	*	*	*	u	*	e	*	*	i	*	a	*	*	*	*	*

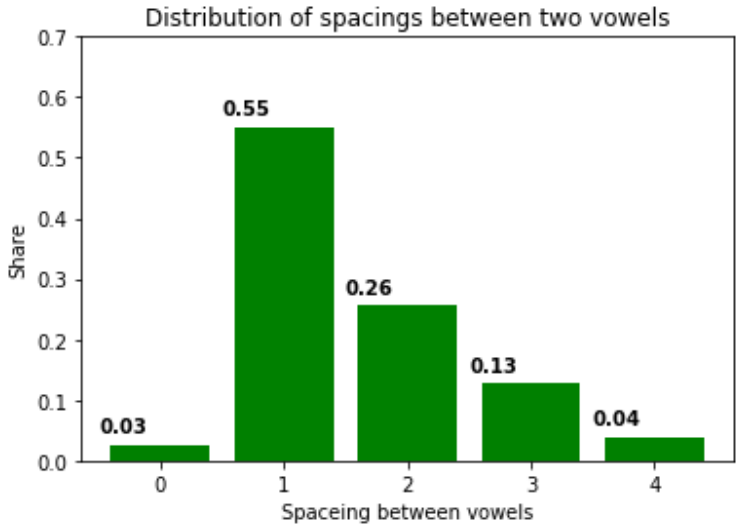


Figure 12: Vowel gap distributions in decrypted text

This gives us new information namely, characters *5,0,4,?,2* are most likely vowels. Of course we need to consider other maps with low sum of square difference as well to see which characters are most likely to be vowels. We analyze the first top 50 character maps with the lowest sum of square difference to see which characters occur in them most often. I obtained this data by counting the number of occurrences of each character and divided it by the total number of characters (namely $50 * 5$ since we have 50 maps and 5 vowels in each). The result is seen in figure 13.

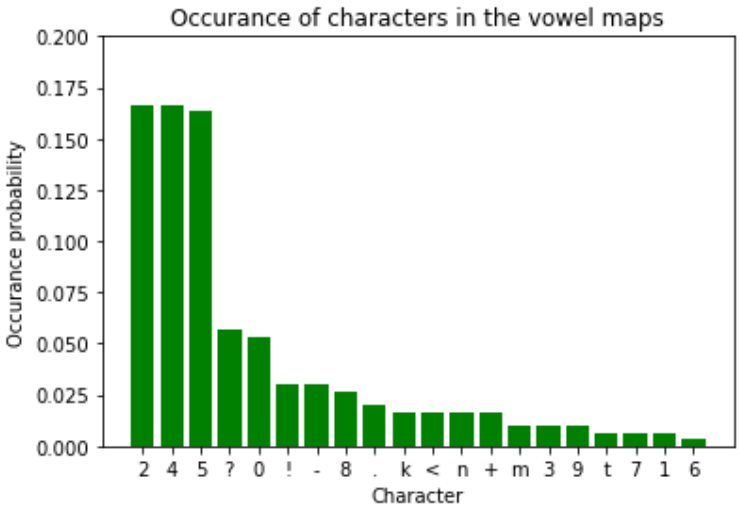


Figure 13: Characters that occur most frequently in potential vowel maps

This leads us to believe that characters *5,0,4,?,2* indeed are most likely vowels.

7 Dictionary attack and manual decryption

The approach with a dictionary attack was made possible due to evidence of word boundaries seen in figure 10.
The dictionary contains all words of present day Slovenian language supplemented with the words from some of Ivan Cankar literary works dating from the early 20th century. We code different filter functions that work by accepting a list of cyphered words and a decryption character map where optional amount of character mappings can be fixed. For cypher text characters of which mappings we don't know, a map to a neutral character (*) is established. At this point we define a term *Word fingerprint* which is used to check if two words could be similar. Two words have the same fingerprint if they are equal in length, characters that have a defined decryption map are in the same spots, characters that appear multiple times in a word and have undefined decryption map must appear under the same letter every time lastly same character must not map to multiple letters. This constrains can be seen clearer in the following examples:
(CT = cypher text, NT = natural text, MP = decryption map, SameFp=Same fingerprint, T=true,F=false)

- $CT = 12345 \quad NT = drevo \quad MP = \{\} \quad SameFP = T$
- $CT = 12342 \quad NT = drevo \quad MP = \{\} \quad SameFP = F$
  (because "2" maps to two different characters namely "r" and "o")
- $CT = 12342 \quad NT = drevr \quad MP = \{\} \quad SameFP = T$
- $CT = 12342 \quad NT = drevr \quad MP = \{"2" : "m"\} \quad SameFP = F$
  (because "2" would map to "r" although "2"—>"m" mapping is already determined)
- $CT = 12342 \quad NT = dmevm \quad MP = \{"2" : "m"\} \quad SameFP = T$
- $CT = 12342 \quad NT = dmemm \quad MP = \{"2" : "m"\} \quad SameFP = F$
  (because "4" maps to "m" although "m" is already mapped to)
- $CT = 12342 \quad NT = dmemm \quad MP = \{"2" : "m", "4" : "m"\} \quad SameFP = F$
  (because 2 characters map to same letter)

The filter function then searches the dictionary by the following procedure:

1. Search for words with same length as cyphered words
2. From those filter out those that have same fingerprint as cyphered words when no additional decryption map is given
3. If decryption map is given filter out the words that satisfy the presented map

With these tools it is easy for the user to define a new map and potential words and see which words are available in the language dictionary.
From all the data gathered so far and with the tools developed, I was able to find a decryption map that shows some promise (seen below).

Table 3: Promising partial decryption map

+	6	7	n	<	3	5	k	9	m	0	č	4	=	.	?	k	2	t	1	-	!	8
*	*	*	*	*	*	a	*	n	*	u	j	e	l	*	i	*	o	*	t	*	v	*

Text yielded by the promising partial decryption map:

tvoje*i*lo**neje***no*il
teakotije*ela*olo*o*e
*o**jiliv*aj*anknaj*anjev
*ali*ov*ajla*e*ola*a*o
ako*o**je*lona*takojkjen
*elne**i*to*u*i*vnkute*a
*nelokla*uv*e*u*jetonaj
v*aj*anknaj*enjevakotije
lo*o*eatakoj

In the deciphered text we can see some words and phrases that indicate that we might be going in the right direction:
tvoje, ako ti je (common phrase from the early 20th century)**, tako** or **takoj,naj**.

8 Fully automated approach

In this section we explore an option for automatic decryption of custom cypher texts using extensive natural language corpora. For English language a corpora with around trillion words is available as are already preprocessed files containing a collection of words, trigrams and bigrams and their counts. For Slovenian language no such data have been found yet, that is why we will initially focus on deciphering English cryptograms.
The main idea in this approach is to exploit word, bigram and trigram frequencies of the language to implement a local optimization and text segmentation algorithms.

8.1 Hill climbing

The hill climbing approach consists of 3 main building blocks. Namely an initial encoded text X, an evaluation function F and a procedure to find local neighbors of X. In our problem we use the logarithm of a probability for finding same set of trigrams as in string X in natural language. This is computed by taking the logarithm of a product of trigram frequencies of individual trigrams found in string X. Mathematical expression seen below. n_i is number of occurrences of some trigram in the corpus N is a number of all the trigrams in a corpus and j is just the end index of a trigram list constructed from the string X.

$$F = \log_{10} \prod_{i=0}^{j} \left(\frac{n_i}{N} \right)$$

We want our algorithm to find a string from a set of neighboring strings to the string x that will have the maximal value of F.
It is now time to define what a neighboring string is. For each string X we can construct an ordered set of bigrams, where less probable bigrams will be at the beginning of the list. Since we want to maximize F it is clear that by having more probable bigrams in this list we will also have more probable trigrams and thus higher values of F. The generation of a list of neighboring strings is then defined as a procedure where we try to replace 20 least probable bigrams with some other bigram. The hill climbing algorithm then starts evaluating neighbors and replaces the current string X with a neighboring string X_neigh if $F(X_neigh) > F(X)$. We can restart the algorithm arbitrary number of times and randomize

character map each time to locally maximize F at different points of the search space. When the algorithm ends we are presented with a list of locally optimized candidates

8.2 Choosing the candidate and text segmentation

For selecting an optimal candidate and segment the string into individual words we exploit word frequencies in the corpus. For each candidate we create a set of all possible string segmentations with a maximal word length of 20. These segmentations are essentially lists of words, for which we can check how frequent they are in the corpus and then taking the product of all their frequencies to evaluate the probability of some segmentation. For words that we cant find in the corpus we define a probability of this words existence. This probability is dependent on the length of the word. Longer words are going to be less probable. The probability is defined below, where N is the number of all words in the corpus.

$$\frac{10.0}{(N \times 100^{len(word)})}$$

Mathematical expression for evaluating segmentations is defined below. Where P is probability of some segmentation N is the number of words in the corpus n_i is the number of occurrences of a certain word and j is the end index of a list of words in a segmentation.

$$P = \prod_{i=0}^{j} \left(\frac{n_i}{N} \right)$$

The segmentation algorithm then returns the values of segmentations (segmented strings and their probabilities) for each candidate found by the hill climb procedure and we select the one with the highest P value to be the decoded text.

8.3 Examples

To test our algorithm we created an encoding function which accepts a permuted alphabet as a key for encoding and returns the encoded text. The hill climbing algorithm accepts the encoded text as a single string with no spaces. In the examples below an original text, a key, encoded text and the output of our algorithm is provided.

1. Long wikipedia text Original string:
   egyptiantempleswerebuilttocommemoratethepharaohsandtosupportthecentralfunctio
   nsoftheirreligiongivingofferingstothegodsreenactingtheirmythologicalinteractionsthrou
   ghfestivalsandwardingofftheforcesofchaosritualsitwasbelievedinvokedthedivinepresenc
   esustainedthegodandenabledittocontinuetoupholdthedivineorderoftheuniversetemples
   wereimportantreligioussitesforallclassesofegyptianseventhoughmostpeoplewereforbidd
   enfromenteringtheirmostsacredareastheyareamongthelargestandmostenduringexample
   sofegyptianarchitecturewiththeirelementsarrangedanddecoratedaccordingtocomplexp
   atternsofreligioussymbolismalargetemplecouldownsizabletractsoflandandemploythous
   andsoflaymentosupplyitsneedssometemplessuchasabusimbelhavebecometouristattract
   ionsthatcontributesignificantlytothemodernegyptianeconomyegyptologistscontinuetos
   tudythesurvivingtemplesfortheirinvaluablesourcesofinformationaboutancientegyptian
   society
   Key:cosfmlejypqzxrvibuhtwagnkd
   Encoded string:
   mekitycrtmxizmhgmumowyzttvsvxxmxvuctmtjmijcucvjhcrftvhwiivuttjmsmrtuczlwrs

tyvrhvltjmyuumzyeyvreyayrevllmuyrehtvtjmevfhummrcstyretjmyuxktjvzveysczyrtm
ucstyvrhtjuvwejlmhtyaczhcrfgcufyrevlltjmlvusmhvlsjcvhuytwczhytgchomzymamfyra
vqmftjmfyayrmiumhmrsmhwhtcyrmftjmevfcrfmrcozmfyttvsvrtyrwmtvwijvzftjmfyay
rmvufmuvltjmwryamuhmtmxizmhgmumyxivutcrtumzyeyvwhhytmhlvuczzszchhmhv
lmekitycrhmamrtjvwejxvhtimvizmgmumlvuoyffmrluvxmrtmuyretjmyuxvhthcsumfcu
mchtjmkcumcxvretjmzcuemhtcrfxvhtmrfwuyremncxizmhvlmekitycrcusjytmstwumgy
tjtjmyumzmxmrthcuucremfcrffmsvuctmfcssvufyretvsvxizmnicttmurhvlumzyeyvwhhk
xovzyhxczcuemtmxizmsvwzfvgrhydcozmtucsthvlzcrfcrfmxizvktjvwhcrfhvlzckxmrtvh
wiizkythrmmfhhvxmtmxizmhhwsjchcowhyxomzjcamomsvxmtvwuyhtcttucstyvrhtjcts
vrtuyowtmhyerylyscrtzktvtjmxvfmurmekitycrmsvrvxkmekitvzveyhthsvrtyrwmtvhtw
fktjmhwuayayretmxizmhlvutjmyuyraczwcozmhvwusmhvlyrlvuxctyvrcovwtcrsymrtm
ekitycrhvsymtk
Decoded string:
e g y p t i a n t e m p l e s w e r e b u i l t t o c o m m e m o r a t e t h e p h a r a o
h s a n d t o s u p p o r t t h e c e n t r a l f u n c t i o n s o f t h e i r r e l i g i o n g
i v i n g o f f e r i n g s t o t h e g o d s r e e n a c t i n g t h e i r m y t h o l o g i c
a l i n t e r a c t i o n s t h r o u g h f e s t i v a l s a n d w a r d i n g o f f t h e f
o r c e s o f c h a o s r i t u a l s i t w a s b e l i e v e d i n v o k e d t h e d i v i n
e p r e s e n c e s u s t a i n e d t h e g o d a n d e n a b l e d i t t o c o n t i n u e
t o u p h o l d t h e d i v i n e o r d e r o f t h e u n i v e r s e t e m p l e s w e r e
i m p o r t a n t r e l i g i o u s s i t e s f o r a l l c l a s s e s o f e g y p t i a n s e
v e n though most people were forbidden from entering their most sacred areas they
are among the largest and most enduring examples of egyptian architecture with their
elements arranged and decorated according to complex patterns of religious symbolism
a large temple could own sizable tracts of land and employ thousands of laymen to
supply its needs some temples such as abu simbel have become tourist attractions
that contribute significantly to the modern egyptian economy egyptologists continue
to study the surviving temples for their invaluable sources of information about ancient
egyptian society

2. Shorter wikipedia text: Original string:
thesunisthestaratthecenterofthesolarsystemitisanearlyperfectsphereofhotplasmawithin
ternalconvectivemotionthatgeneratesamagneticfieldviaadynamoprocess
Key:tblzrfcuekyohgivmnapqsjdxw
Encoded string:
puraqgeapuraptntppurlrgprnifpuraiotnaxaprhepeatgrtnoxvrnfrlpavurnrifuipvotahtjep
uegprngtoligsrlpesrhipeigputpcrgrntprathtcgrpelferozsettzxgthivnilraa
Decoded string:
the sun is the star at the center of the solar system it is a nearly perfect sphere of hot
plasma with internal convective motion that generates a magnetic field via a dynamo
process

3. 50 character custom text
Original string:
thisdogisfarawayfromhomeweshouldtrytofindhisowners
Key: shgnbvdqypmfozxwajilrkcute
Encoded string:
lqyinxdyivsjscstvjxoqxobcbiqxrfnljtlxvyznqyixczbji
Decoded string:
chat se patrol on our le whew in i them bsc luce rads hat end ilt

4. 83 character custom text
   Original string:
   thisdogisfarawayfromhomeweshouldtrytofindhisownersmaybetheywillevengiveussome reward
   Key: shiorwmzvcxdanqklyebgjfutp
   Encoded string:
   bzveoqmvewsysfstwyqazqarfrezqgdobytbqwvnozveqfnryeasthrbzrtfvddrjrnmvjrgeeqar yrfsyo
   Decoded string:
   c his to dispar away prom home wes hoult cry cop in this owners maybe chey will even dive us some re wart

5. 119 character custom text:
   Original string:
   thisdogisfarawayfromhomeweshouldtrytofindhisownersmaybetheywillevengiveussome rewardwiththemoneywegetwecouldbuyourowndog
   Key: nuxiwjasomeqvpkbhdfgcrtylz Encoded string:
   gsofikaofjndntnljdkvskvwtwfskcqigdlgkjopisofktpwdfvnluwgswltoqqwrwpaorwcffkvw dwtnditogsgswvkpwltwawgtwxkcqiuclkcdktpika Decoded string:
   thi coal icf urum us frap hape mech and otrs ta fibo hi camber cpus we the smid dev eb live nc cape re muro mith the pabes me let megan down san rambo al

6. 177 character custom text: Original string:
   thisdogisfarawayfromhomeweshouldtrytofindhisownersmaybetheywillevengiveussome rewardwiththemoneywegetwecouldbuyourowndogithinktakingcareforadogwillstrenght enourrelationshipaswell Key: rmgtauhcvinzksoelwyfjxqpbd
   Encoded string:
   fcvytohvyurwrqrbuwokcokaqaycojztfwbfouvstcvyoqsawykrbmafcabqvzzaxashvxajyyo kawaqrwtqvfcfcakosabqahafqagojztmjbojwoqstohvfcvsnfrnvshgrwauowrtohqvzzyfwas hcfasojwwazrfvosycveryqazz
   Decoded string:
   this dog is faraway from home we should try to find his owners maybe they will even give us some reward with the money we get we could buy our own dog i think taking care for a dog will strenghten our relationship as well

   From the examples it is clear that the method works for long enough strings. We see that decoding is not correct when cypher text is 50,83 and 119 characters long. The first decoding is successful but there are some problem with segmentation thus the single letters at the beginning.

9 Future plans

An important goal for the future concerning the Hill climbing approach is to create a similar corpus for Slovenian language and use it with the introduced algorithm. We will also explore ways to improve the current algorithm for faster and more reliable decoding (use analytical data produced by the tools also described in this work). Next goal is to implement responsive graphic interface for the user to manipulate decryption maps. Next step is providing the user with real time visual data about the probability of his/hers pereferred (words still available in the dictionary given the map, vowel gap distribution, index of coincidence).

We plan to implement an automated system to predict which character mappings are more probable based on all statistics combined and combine that with the user interface in such way that when a user clicks on a not yet mapped character he will see a table of most probable maps.
We also want to try out a different approach involving neural networks. The idea is to generate many examples of random substitution cypher cypher texts from a fragment of known Slovenian text approximately the same length as Elizika letter. First we would try this with text which have word separation.
When we have the database of cypher texts, corresponding maps and known text I could feed this data to the neural net. The heuristic for the neural network to follow would be the number of words we can find in a dictionary for a decrypted text. More words we can find the more successful was the network.

10 Conclusions

Although the decryption of the letter was not yet completely successful we managed to make quite some interesting observations about the Elizika letter. But more importantly useful analytical, decryption manipulating and automation tools were developed. These will ease further attempts to decipher this letter and other substitution cyphers.

Sources

- Beautiful Data: Toby Segaran and Jeff Hammerbacher
- Bachelors thesis: Sašo kodrič, Orodja za razbijanje substitucijske šifre, Ljubljana 2013
- Corpora of Cankar literature: http://lit.ijs.si/leposl.html
- Stamp picture: http://www.old-stamps.com/stamps/kaiserlich-koenigliches-oe franciscus-josephus-1908-kaiserliche-koenigliche-oesterreichische-post 3144.html
- Index of coincidence: https://en.wikipedia.org/wiki/Index_of_coincidence
- The Black Chamber: http://www.simonsingh.net/The_Black_Chamber/
- German letter frequency: http://practicalcryptography.com/cryptanalysis/ letter-frequencies-various-languages/german-letter-frequencies/
- Hungarian letter frequency: https://en.wikipedia.org/wiki/Hungarian_ alphabet
- Italian letter frequency: https://everything2.com/title/Letter+frequency+ in+several+languages