Decrypting Cipher text Using MCMC

Eitan Zimmerman, Yonatan Lourie

Department of Statistics, The Hebrew University of Jerusalem

July 2022

1. Introduction

We look into how to break classical ciphers using Markov Chain Monte Carlo (MCMC) techniques. Simple substitution ciphers have previously been cracked using the MCMC algorithm[1]. In this article, we implemented the algorithm ourselves, and we were able to implement it on the Hebrew language. The fact that the Hebrew language is a morphologically rich language (MRL), makes the linguistic challenge harder[2] than “easier” languages like english.

1.1 Classical ciphers

Encoding data is the process of encryption in cryptography. This technique transforms the information's initial plaintext representation into an alternate version known as ciphertext. Today, there are many types of encryption methods which we will discuss in this paper. We will focus only on classical ciphers, and particularly with substitution ciphers.
In a substitution cipher, letters are consistently swapped out for other letters throughout the message. The Caesar cipher is a well-known example of a substitution cipher. Each letter in a message is swapped out for a letter three positions later in the alphabet to encrypt it using the Caesar cipher. As a result, A is changed to D, B to E, C to F, etc. As an illustration, "STATISTICS" is encoded as "VWDWLVWLFV".

[1] S. Conner (2003), Simulation and solving substitution codes.
[2] Reut Tsarfaty, Amit Seker, Shoval Sadde, and Stav Klein. (2019). What’s wrong with Hebrew NLP? and how to make it right

1.2 Problem definition

Suppose $A \ni \{Alphabet\}$ (ordered letters), T text that we wish to encode, which all of his letters subset of A. Let's also supposed $A^{'}$ wit the same alphabet as i n A but with a different order. Thus, we can construct a bijection $F\colon\ A \to A^{'}$, apply this function to T, to get an encrypted text $\widetilde{T}$: $F(T)\ = \widetilde{T}$.

Hence to encode a message all that need to be done is to find some permutation σ to the relevant character set, and then the act of decoding reduces to the problem of finding the inverse σ^{-}.

Given the explanation above, our "True" text can be viewed as a sequence of characters generated by some Markov Chain. That is, if we were able to assign a probability to any possible character transition, then we could calculate the likelihood of any possible permutation function by applying it on our text and multiplying the resulting sequence transition probabilities.

The transition probabilities could be based on any n-grams, in our case we used 2-grams, which is also known as bigram.

Having this breakdown in mind, the most simple, naive solution will be looping on all possible permutations, applying each one on the text, calculating its likelihood based on our n-grams transition probabilities and taking the one with the highest probability. It can be easily shown that this kind of solution will become infeasible as our character set grows as the number of permutations is $|S_n|\ =\ n!$. For the English language it will be equal to $26!$ and for Hebrew $27!$.

As with any problem that can be shown to have Markov Chain structure, we can take advantage of MCMC methods to “random walk” on our permutations infeasible set, this chain will have an equilibrium distribution as the posterior distribution of the true permutation given the encoded message. Running the MCMC algorithm for a large number of iterations will hopefully get us close to this distribution and allow us to sample a close solution from it.

2. Methods

2.1 Constructing the Transition Matrix

Before considering any MCMC algorithm to solve our problem the character set needs to be decided on. As this method for solving classical ciphers has been tested by many before, we wanted to try our own implementation for Hebrew language ciphers.

We used the “wikipedia” Hebrew archive text file to create our transition probabilities, in part of our preprocessing we left a character set size of 28 - 27 characters in Hebrew alphabet and one more character for “space”.

Part of our preprocessing stage was to remove any unwanted characters from our corpus so a clean text file is left for our transition matrix built.

Plot for the transition matrix probabilities can be seen in Appendix A.

2.2 General explanation of the Metropolis-Hastings algorithm

One popular implementation of the MCMC approach for approximate inference is Metropolis-Hastings. It makes it possible to sample from a probability distribution when direct sampling is challenging, typically because an intractable integral is present.

A proposal distribution $q(\hat{\theta}|\theta)$ is used in Metropolis Hasting to derive a parameter value. We then compute a ratio to decide whether $\hat{\theta}$ is approved or denied: $\frac{p(\hat{\theta}|D)}{p(\theta|D)}$. The next step is to select a random number $r \in (0, 1)$ and determine if it falls inside the ratio or not. If you agree, set $\theta \hat{=} \theta_i$ and keep going.

By the time we're done, we have a sample of θ values that we can utilize to create quantities over a rough posterior, including the expectation and uncertainty bounds. In reality, we often have a warm-up phase to eliminate bias towards initialization values and a period of tweaking to attain an acceptable acceptance ratio for the algorithm.

2.3 Simulation implementation

Having our language model (e.g transition probability) we are still missing the update mechanism (algorithm). As we are using the Metropolis-Hasting the only thing left to take care of is the calculation of suitable proposal and acceptance probabilities. As discussed in our problem definition section, we can use the multiplication of any permutation transition probabilities to identify its equivalent likelihood.

To be more explicit, the likelihood of any permutation sigma will be calculated:

$$L(\sigma) \; = \; \beta(\sigma_{t0}) \prod_{k=1}^{n} \gamma(\sigma_{t(k-1)}, \sigma_{t(k)})$$

Where

$\beta(\sigma)$ - is the prior probability that the initial letter of the decoded text is the permutation initial letter.

$(\sigma_{t(k-1)}, \sigma_{t(k)})$ - The prior probability that the kth letter is $\sigma_{t(k)}$ given that the (k-1)th letter is $\sigma_{t(k-1)}$

Having the likelihood of any permutation we can decide on an acceptance probability. As with many other MCMC techniques we can use the likelihood ratio $L(\sigma_{proposal})/L(\sigma_{current})$ together with randomly drawn from uniform distribution probability to decide whether to accept or reject the proposal.

The algorithm break down for the chain is as follows:

1. Calculate the transition matrix based on wikipedia corpus.
2. Propose new random permutations

3. Calculate the likelihood of both current and proposed permutations.
4. Calculate the acceptance probability
5. Accept based on 4 and some random probability drawn from Uniform distribution. This allows our chain to have some degree of freedom.

3. Results

First, let's see an example of the algorithm.

<u>The original message</u>:

המלחמה ברצועת עזה נגמרה לגמרי ושלום עולמי קיים בארץ ישראל. אף על פי כך, נדקר טיפוס אחד, המצב

בשווקים הידרדר משמעותית בחודש האחרון בעקבות המצב החמור במושבה מאדים.

<u>The encrypted message</u>:

נפדנ ףןקץזסדףסו ףבכנק ףפכנקיףזאפזשףסזפניףטיישףןעקחףיאקעפףעהףספףתיףרלףבצטקףסיתזגףעדצף

נעןףןאזזטישף יצקצקףנאנסזדידףןדזצאף עדקזמףןסטןזדף נעןף דנזקףןנזאן ףנעציש.

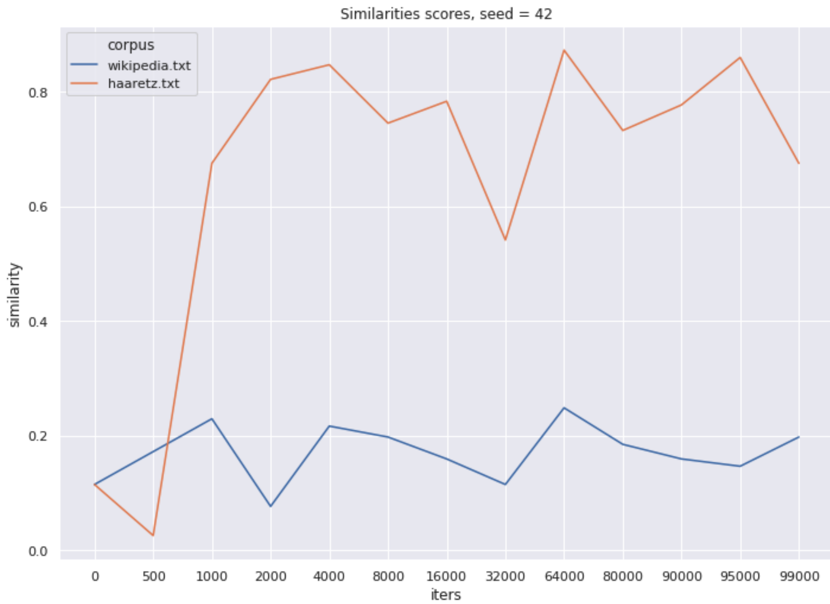
<u>Decryption by the MCMC procedure</u>:

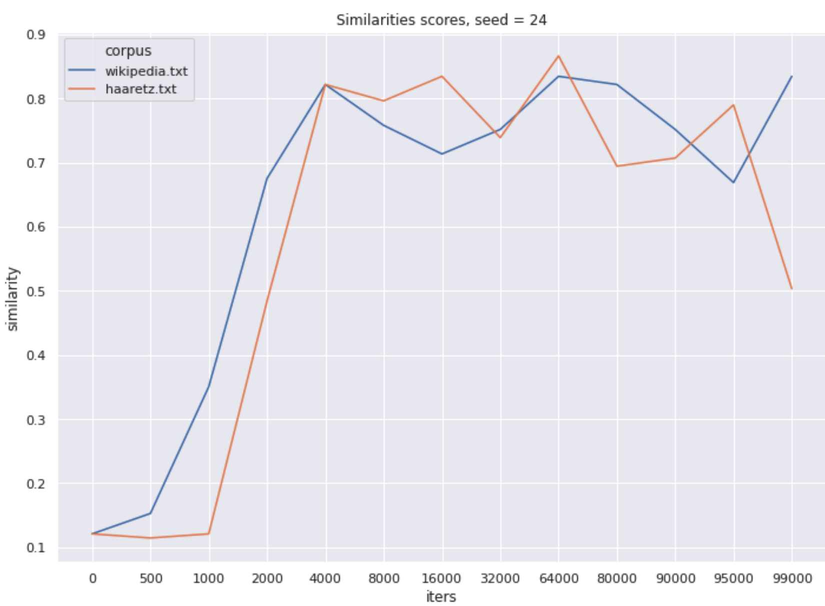
המלחמה ברצועת עזה פגמרה לגמרי ושלום עולמי דיים בארץ ישראל אף על ני כך פקדר סינוט אחק המצב בשוודים

היקרקר משמעותית בחוקש האחרון בעדבות המצב החמור במושבה מאקים.

The convergence process by iterations to the real text is in Appendix B. (we also added the convergence when we use english cipher with the same algorithm).

The transition matrix is calculated on the Haaretz corpus, with 100000 iterations.

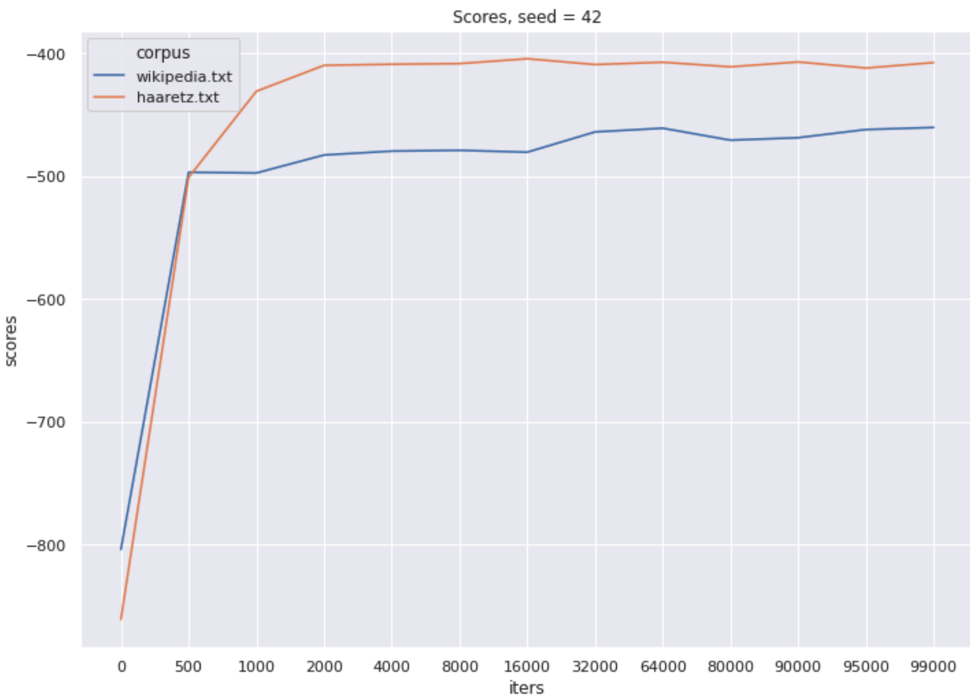


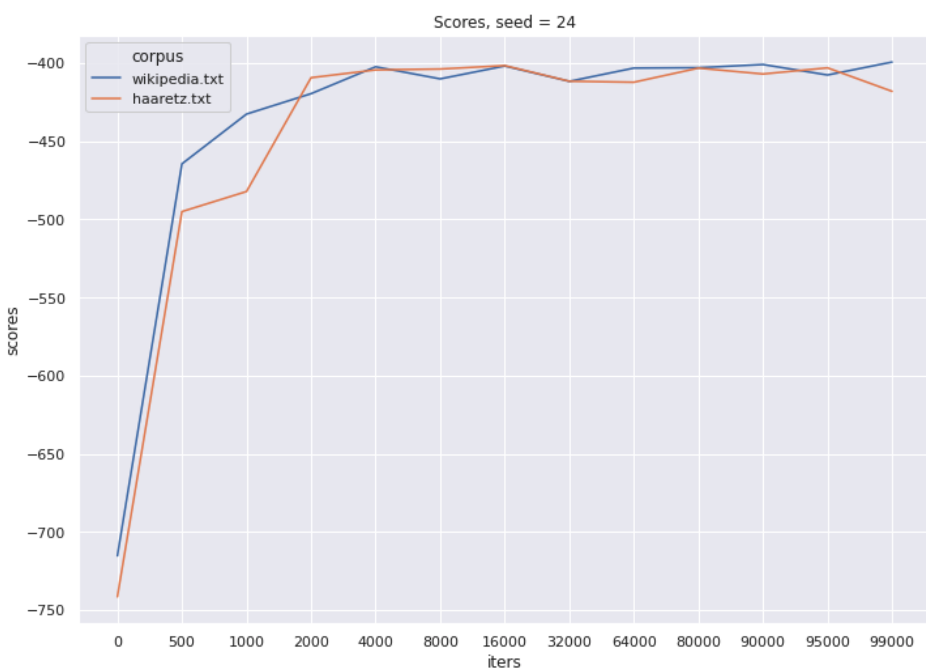


From the following graphs, It can be seen that the choice of the corpus is very significant. But, when we tried to run the program with a different seed, we got a bit different results and it seems like the similarity converges to similar numbers.

When we try more messages we figure that this algorithm can't "crack" any code, and it depends on the context, the order of letters, and the size of the cipher.

When we look at the scores ($log(\sum)$) over all of the current iteration probabilities, we can see that the trend is much more stable. We can see the "warm up" of the MCMC at the first iterations, and the convergence in the last iterations.





4. Discussion

When we tried the algorithm with english text, the convergence was much higher and smoother, and the variance with dependence on the seed is much lower. The problem with processing Hebrew text (NLP) is much harder than English due to Nikud, multiple meanings of the same word, and more. An interesting research question could be to implement the same algorithm but this time with Nikud letters as part of the character set and maybe try to implement it with higher n-gram (like 3 or 4).

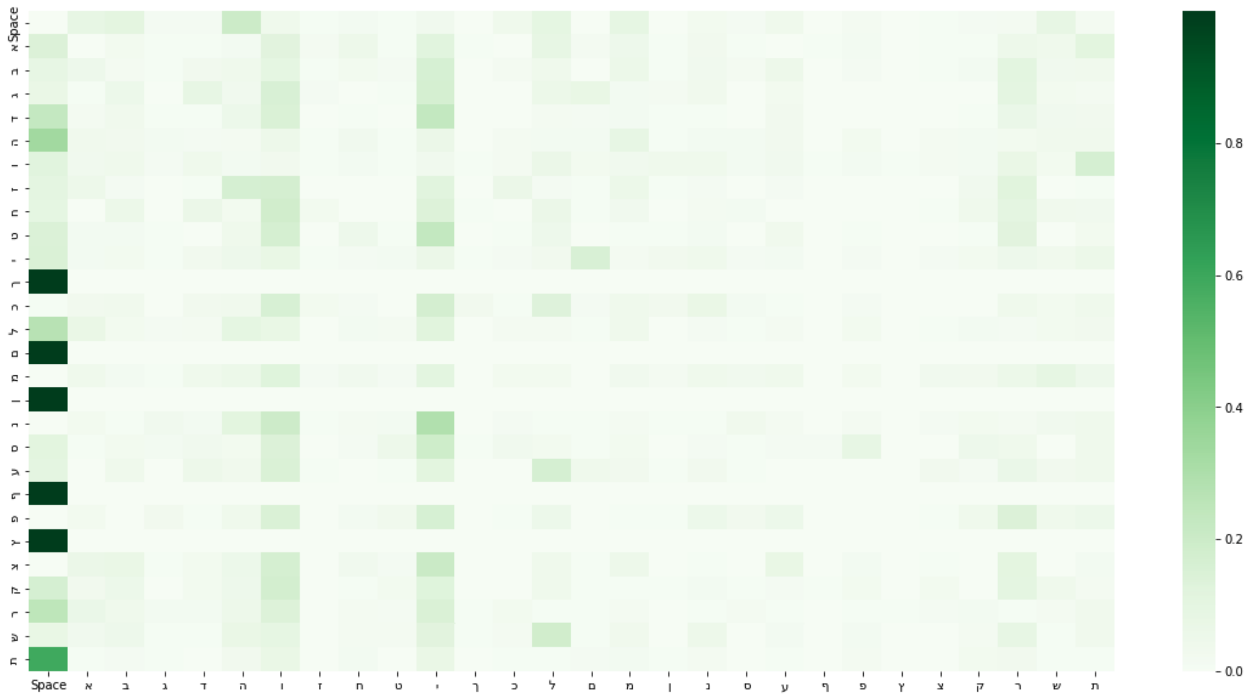
There are much better solutions to tackle the problem of decrypting classical ciphers, so this solution is not that useful. But this is a really good example on how we can implement the MCMC procedure to solve a non-trivial problem.

References

[1] S. Connor. *"Simulation and Solving Substitution Codes"*. In: Master's Thesis, Department of Statistics, Uni 46.1 (2003), pp. 179–205.

[2] J. Chen and J. S. Rosenthal. *"Decrypting Classical Cipher Text Using Markov Chain Monte Carlo"*. In: Statistics and Computing 22.1 (2012), pp. 397–413.

Appendix A. Transition matrix



Appendix B. Convergence of MCMC process

iter: 0 טוהכוטופדץסמנןמקטוםפודטןהפודיןסרהסשןמסהויןתיישןפ דדןירך הן עןמהןזיןחגןםבתדןליזסאן כבןטוץפןפרססתישןטיבדבדןורומסנינןפכסברןט כרסצןפמתפסנןטוץפןטכוסדןפוסרפטון ביש

iter: 0 טוהכוטופדץסמנןמקטוםפודטןהפודיןסרהסשןמסהויןתיישןפ דדןירך הן עןמהןזיןחגןםבתדןליזסאן כבןטוץפןפרססתישןטיבדבדןורומסנינןפכסברןט כרסצןפמתפסנןטוץפןטכוסדןפוסרפטון ביש

iter: 2 טוהכוטופדץסמנןמקטוםחודטןהחודיןסרהסשןמסהויןתיישןפ דדןירך הן עןמהןזיןףגןםבתדןליזסאן כבןטוץפןפרססתישןטיבדבדןורומסנינןפכסברןט כרסצןפמתפסנןטוץפןטכוסדןפוסרפטון ביש

iter: 3 טוהכוטןשדץסמנןמקטוםחודטןהחודיןסרהספןמסהויןתייפןש דדןירך הן עןמהןזיןףגןםבתדןליזסאן כבןטוץשןשרססתיפןטיבדבדןורומסנינןשכסברןט כרסצןשמתשסנןטוץשןטכוסדןשוסרשטון ביפ

iter: 4 טוהכוטןשדעסמנןמקטוםחודטןהחודיןסרהספןמסהויןתייפןש דדןירך הן ץןמהןזיןףגןםבתדןליזסאן כבןטועשןשרססתיפןטיבדבדןורומסנינןשכסברןט כרסצןשמתשסנןטועשןטכוסדןשוסרשטון ביפ

iter: 3000 המלחמה ברגועת עזה פצמרה לצמרי ושלום עולמי דיים בארץ ישראל אן על ני כך פקדר סינוט אחק המגב בשוודים היקרקר משמעותית בחוקש האחרוף בעדבות המגב החמור במושבה מאקים

iter: 13000 המלשמה ברצועת עזה טגמרה לגמרי וקלום עולמי דיים בארץ יקראל אן על ני ספ טחדר כינוף אשה המצב בקוודים היחרחר מקמעותית בשוחק האשרוך בעדבות המצב השמור במוקבה מאחים

iter: 14000 המלשמה ברכועת עזה פגמרה לגמרי ואלום עולמי דיים בחרץ יארחל חך על ני צף פקדר סינוט חשק המכב באוודים היקרקר מאמעותית בשוקא החשרון בעדבות המכב השמור במואבה מחקים

iter: 18000 המבדמה לרחוקת קנה פגמרה בגמרי ואבום קובמי ציים לכרץ יארכב כך קב עי סט פשצר זיעון כדש המחל לאווצים הישרשר מאמקותית לדושא הכדרוף לקצלות המחל הדמור למואלה מכשים

iter: 22000 המלחמה ברגושת שנה סטמרה לטמרי ועלום שולמי ציים בארץ יעראל אף של פי כז סקצר דיפוך אחק המגב בעווצים היקרקר מעמשותית בחוקע האחרון בשצבות המגב החמור במועבה מאקים

iter: 43000 המדשמה לרכואת אזה נגמרה דגמרי ועדום אודמי פיים לקרץ יערקד קף אד טי חס נבפר ציטון קשב המכל לעווּפים היברבר מעמאותית לשובע הקשרוך לאפלות המכל השמור למועלה מקבים

iter: 52000 השלקשה ברחואת אזה צפשרה לפשרי ועלום אולשי ניים במרץ יערמל מס אל כי גף צדנר טיכוך מקד השחב בעוונים הידרדר שעשאותית בקודע המקרון באנבות השחב הקשור בשועבה שמדים

iter: 69000 המלשמה ברצועת עזה פנמרה לנמרי וקלום עולמי דיים בארץ יקראל אס על טי כך פחדר גיטוף אשה המצב בקוודים היחרחר מקמעותית בשוחק האשרון בעדבות המצב השמור במוקבה מאחים

iter: 72000 המלאמה ברצועת עזה כסמרה לסמרי וקלום עולמי דיים בשרץ יקרשל שך על ני פט כחדר גינון שאה המצב בקוודים היחרחר מקמעותית באוחק השארוף בעדבות המצב האמור במוקבה משחים

iter: 99000 המלקמה ברצואת אזה סכמרה לכמרי ועלום אולמי גיים בשרץ יערשל שך אל ני חן סטגר פינוף שקט המצב בעווגים היטרטר מעמאותית בקוטע השקרוד באגבות המצב הקמור במועבה משטים

The original message:

המלחמה ברצועת עזה נגמרה לגמרי ושלום עולמי קיים בארץ ישראל. אף על פי כך, נדקר טיפוס אחד, המצב בשווקים הידרדר משמעותית בחודש האחרון בעקבות המצב החמור במושבה מאדים.

In english

iter: 1 IKQEUPZUPBNYREQOEUQRTUEKTURPFKEZUYGUBQOUTBQOJPNQEPYOUGRYBUNRTWV PJTZUQO UTCVQSPEAUYGUJPEPDTOZKPNUQO UQSSUEKTZTUP TQZUOQNYSTYOUKQZURTEQPOT UPOUGVSSUGYRJT

iter: 2000 WHAT IS IFROPTANT APE THE PICHTS OM FAN EFANLIRATION MPOF RPEXUGILES ANG EBUADITY OM LITIZENSHIR ANG ADD THESE IGEAS NARODEON HAS PETAINEG IN MUDD MOPLE

iter: 6000 WHAT IS IMBORTANT ARE THE RIKHTS OF MAN EMANGIBATION FROM BREPUDIGES AND EQUALITY OF GITIVENSHIB AND ALL THESE IDEAS NABOLEON HAS RETAINED IN FULL FORGE

iter: 13000 WHAT IS IMPORTANT ARE THE RIGHTS OF MAN EMANKIPATION FROM PRECUDIKES AND EQUALITY OF KITIVENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORKE

iter: 40000 WHAT IS IMPORTANT ARE THE RIZHTS OF MAN EMANVIPATION FROM PREBUDIVES AND EJUALITY OF VITICENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORVE

iter: 57000 KHAT IS IMPORTANT ARE THE RICHTS OF MAN EMANGIPATION FROM PREQUDIGES AND EBUALITY OF GITIVENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORGE

iter: 60000 WHAT IS IMPORTANT ARE THE RIZHTS OF MAN EMANVIPATION FROM PRECUDIVES AND EQUALITY OF VITIBENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORVE

iter: 76000 WHAT IS IMPORTANT ARE THE RIZHTS OF MAN EMANCIPATION FROM PREQUDICES AND EKUALITY OF CITIGENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORCE

iter: 79000 WHAT IS IMPORTANT ARE THE RIGHTS OF MAN EMANCIPATION FROM PREQUDICES AND EBUALITY OF CITIVENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORCE

total acceptances: 8819

Original message:

WHAT IS IMPORTANT ARE THE RIGHTS OF MAN EMANCIPATION FROM PREJUDICES AND EQUALITY OF CITIZENSHIP AND ALL THESE IDEAS NAPOLEON HAS RETAINED IN FULL FORCE

Appendix C. Calculation of the transition matrix

- To make the transition matrix, we first needed to get data. We used the MILA[3] resource .(מרכז ידע לתקשוב בשפה העברית)
- From there we took a small portion of the Hebrew wikipedia corpus and from the Haaretz corpus.
- The function that calculates the transition matrix is called `process_text()`

[3] https://yeda.cs.technion.ac.il/index.html