Article

Statistics and Machine Learning Experiments in Poetry

Ovidiu Calin

Department of Mathematics & Statistics, Eastern Michigan University, Ypsilanti, MI 48197, USA; ocalin@emich.edu

Received: 10 June 2020; Accepted: 28 June 2020
First Version Published: 2 July 2020 (doi:10.3390/sci2030048)

Abstract: This paper presents a quantitative approach to poetry, based on the use of several statistical measures (entropy, information energy, N-gram, etc.) applied to a few characteristic English writings. We found that English language changes its entropy as time passes, and that entropy depends on the language used and on the author. In order to compare two similar texts, we were able to introduce a statistical method to asses the information entropy between two texts. We also introduced a method of computing the average information conveyed by a group of letters about the next letter in the text. We found a formula for computing the Shannon language entropy and we introduced the concept of N-gram informational energy of a poetry. We also constructed a neural network, which is able to generate Byron-type poetry and to analyze the information proximity to the genuine Byron poetry.

Keywords: entropy; Kullback–Leibler relative entropy; recurrent neural networks; learning

1. Introduction

This paper deals with applications of statistics and machine learning to poetry. This is an interdisciplinary field of research situated at the intersection of information theory, statistics, machine learning and literature, whose growth is due to the recent developments in data science and technology. This paper topic is important since it contains a quantitative approach to an area that until recently belonged more to the field of arts rather than to the field of exact sciences.

The paper presents first some familiar statistical tools that are based on the usage of letter frequency analysis, which have been used since the beginning of the natural languages theory. The underlying idea is that each language has its own letter distribution frequency, which can be characterized by certain statistical measures. For instance, in English, from the most frequent to least frequent, letters are ordered as *etaoin shrdlu cmfwyp vbgkjq xz*, while in French they are ordered as *elaoin sdrétu cmfhyp vbgwqj xz*. Even within the same language, writers write slightly different, adapting the language to their own style. The statistical tools used in this paper and calculated for specific authors are: entropy, informational energy, bigram, trigram, N-gram, and cross-entropy. Moreover, these statistics may be used to prove or disprove authorship of certain texts or to validate the language of a text.

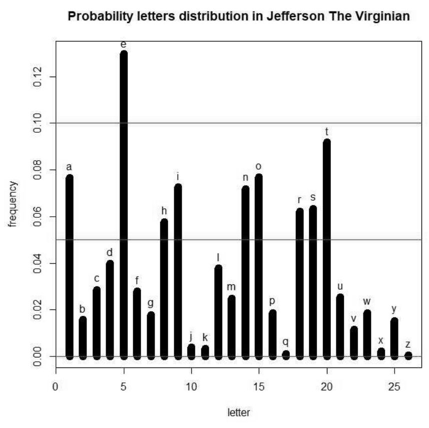
These tools have been useful, as each statistic measures a specific informational feature of the text. However, when it comes to the problem of generating a new text that shares the statistic similarities of a given author, we need to employ recent developments of machine learning techniques. This task can be accomplish by the most recent state-of-the-art advances in character-level language modeling and is presented in the second part of the paper. More specifically, we use recurrent neural networks (RNNs) to generate Byron-type poetry and then to asses their informational deviation from a genuine Byron text. When this difference of similarity becomes small enough, the network training stops and we obtain a generator for Byron poetry. It is worth noting that a similar neural architecture can be employed to generate poetry similar to other authors.

2. Material and Methods

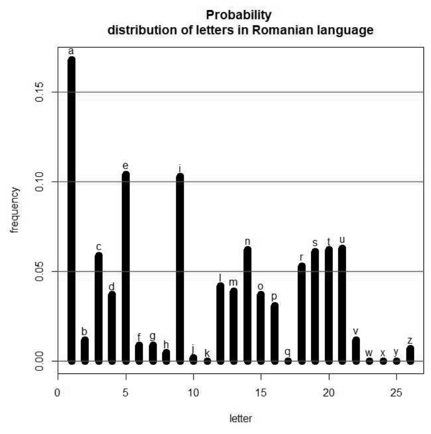
2.1. Letter Frequency

Letter frequency analysis was the first statistical approach to languages. The relative frequency of letters in a text can be transformed into a probability distribution dividing each frequency by the length of the text. When the text is long enough, the Law of Large Numbers assures the convergence of empirical probabilities to theoretical probabilities of each letter.

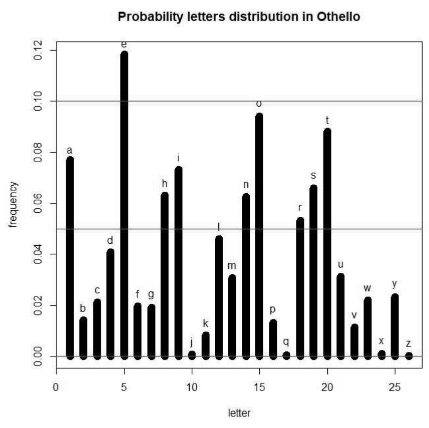
A few letter probability distributions are represented in Figure 1. Parts a and b correspond to English texts, *Jefferson the Virginian* by Dumas Malone and *Othello* by Shakespeare, respectively. We notice a high frequency of letter "e" followed by "t", "a" and "o". In old English (*Othello*) we notice a higher frequency of "o" over "t". Figure 1c,d correspond to Romanian texts, *Memories of Childhood* by Ion Creangă and *The Morning Star*, by Mihai Eminescu, respectively. We notice a high frequency for letter "a" followed by "e", "i", "n" or "r". A bare eye-view inspection of letter probability histograms like the ones in Figure 1 is not very efficient, since besides magnitudes of letter frequency, we cannot infer too much. This is why we need to employ some statistics which characterize these distributions from different perspectives. We shall present a few of these statistics in the next sections. The interested reader can find the 'R' code for the frequency plots online at http://machinelearningofannarbor.org/Frequency.txt.



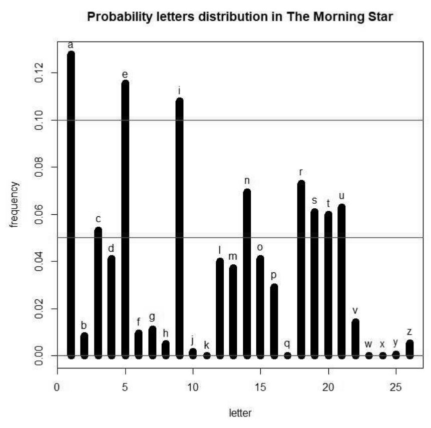
(**a**)



(**c**)



(**b**)



(**d**)

Figure 1. The letter probability distribution for: (**a**). Jefferson the Virgian; (**b**). Othello; (**c**). Memories of Childhood (in Romanian); (**d**). The Morning Star (in Romanian).

2.2. Informational Entropy

The first interesting applications of statistics to poetry had been mainly based on concepts of information theory, such as entropy and informational energy. This way, poetry was treated from the point of view of statistical mechanics. From this point of view, each letter of the alphabet present in a given text was assimilated with a random variable having a certain probability of occurrence.

These probabilities can be approximated empirically by the associated frequencies measured from the text, see Figure 1. According to the Law of Large Numbers, the quotients between the number of occurrences of a letter and the text length tends to the theoretical probability of the letter. Thus, the poem can be modeled by a finite random variable X, which takes letter values $X \in \{a, b, \cdots, z\}$ and is described by the probability distribution

X	a	b	$\cdots$	z
$P(X)$	$p(a)$	$p(b)$	$\cdots$	$p(z)$

This distribution probability table provides the letter frequency structure of the text. Now, each letter is supposed to contribute the text with some "information", which is given by the negative log-likelihood function; for instance, the letter "a" contains an information equal to $-\log_2 p(a)$. The minus sign was considered for positivity reasons, while the logarithm function was chosen for its properties (the information contained by two letters is the sum of their individual information, namely, $-\log_2[p(a)p(b)] = -\log_2 p(a) - \log_2 p(b)$). A weighted average of these letter information is given by the expression of *informational entropy*

$$H = -[p(a)\log_2 p(a) + p(b)\log_2 p(b) + \cdots + p(c)\log_2 p(c)].$$

The statistical measure H represents the amount of information produced, on average, for each letter of a text. It is measured in bits. We recall that one bit is the measure describing the information contained in a choice between two equally likely choices. Consequently, the information contained in 2^N equally likely choices is equal to N bits. It is worth noting that the previous entropy expression mimics the formula introduced in information theory by C.E. Shannon [1] in 1948.

Another interpretation of entropy is the following. Assume that during a text compression each letter is binary encoded as a sequence that consists of 0 s and 1 s. Then, the entropy is the average number of binary digits needed to represent each letter in the most efficient way. Given that the entropy of English is about 4 bits per letter and the ASCII code (Abbreviation from American Standard Code for Information Interchange, which is a character encoding standard for electronic communication) translates each letter into 8 binary digits, it follows that this representation is not very efficient.

The value of the entropy H was related in poetry by the state of organization of the poem, such as rhythm or poem structure. For instance, a poem with a short verse has a smaller entropy, while a poem written in white verse (no rhythm) tends to have a larger entropy. Consequently, a poem has a smaller entropy than a regular plain text, due to its structure. The amount of constraint imposed on a text due to its structure is called *redundancy*. Consequently, poetry has more redundancy than plain text. Entropy and redundancy depend also on the language used, being complimentary to each other, in the sense that the more redundant a text is, the smaller its entropy. For instance, a laconic text has reduced redundancy, since the omission of any word can cause non-recoverable loss of information.

The maximum of entropy is reached in the case when the random variable X is equally distributed, i.e., in the case when the text contains each letter an equal number of times, namely when each letter frequency is $p(i) = 1/26$. The maximum entropy value is given by $H_{\max} = -\sum_{i=1}^{26} \frac{1}{26} \log_2 \frac{1}{26} =$ 4.70 bits per letter.

However, this maximum is far to be reached by any language, due to its own peculiarities, such as preference towards using certain words more often, or bias towards using certain language structures. For instance, in English, there is a high frequency of letter "e" and a strong tendency of letter "t" to be followed by letter "h". Further, "q" is always followed by a letter "u", and so on.

It has been noticed that each poet has its own specific entropy range, due to the usage of favorite words or specific poem structure preferred by the poet. It has also been conjectured that it should be able to recognize the author of a text just from the text entropy. After the examination of several examples we are now certain that it is easier to recognize the poetry language easier than the poet itself.

Our experiments have shown that the entropy range for English language is between 4.1 and 4.2 bits per letter. Other languages have a different range. For instance, the Romanian language has the range between 3.7 to 3.9 bits per letter. The fact that English has more entropy is due to the fact that the probability distribution of letters in English is more evenly distributed than in Romanian, fact that can be also seen from Figure 1 after a careful look.

In the following, an entropy table is provided containing a few characteristic English writings.

Title	Entropy
Jefferson the Virginian	4.158
Nietzsche	4.147
Genesis	4.105
Macbeth	4.187
Romeo and Juliet	4.182
Othello	4.160
King Richard III	4.214
Byron vol. I	4.184
Byron vol. II	4.184

It is worth noting that Byron's first poetry volume and Byron's second volume (containing letters and journals) have the same entropy, 4.184 bits, which shows consistency across the works of this writer. We also note that Shakespeare's works (Macbeth, Romeo and Juliet, Othello, King Richard III) contain a larger variation than Byron's, taking values in the range 4.1–4.2 bits. The entire text of *Jefferson the Virginian* has an entropy of 4.158 bits (quite close to the entropy of 4.14 bits found by Shannon considering just a part of this work). The English translation of the works of the German philosopher Nietzsche (1844–1900) has an entropy of 4.147 bits. The lowest entropy occurs for the Bible chapter on *Genesis*, 4.105 bits, a fact explained by a strong tendency in repetition of certain specific biblical words.

For the sake of comparison, we shall provide next an entropy table containing a few Romanian authors:

Author	Entropy
Eminescu	3.875
Ispirescu	3.872
Creanga	3.877

We have considered Eminescu's famous poem, *Luceafarul (The Morning Star)*, Ispirescu's fairy tales and Crangă's *Amintiri din Copilărie (Memories of Childhood)*. It is worth noting that all these writings have entropies that agree up to two decimals -3.87 bits. We infer that the entropy of Romanian language is definitely smaller than the entropy of English. One plausible explanation is the rare frequency of some letters in Romanian language, such as "*q*", "*w*", "*x*", and "*y*", comparative to English.

However, there are also some major limitations of entropy as an assessing information tool. First of all, the entropy does not take into account the order of letters. This means that if we scramble the letters of a nice poem into some rubbish text, we still have the same entropy value. Other questions have raised, such as whether the random variable X should include in its values blank spaces and punctation signs, fact that would definitely decrease the information entropy value.

Even if entropy is a robust tool of assessing poems from the point of view of the information content, it does not suffice for all purposes, and hence it was necessary to consider more statistical measures for assessing the information of a text.

2.3. Informational Energy

If information entropy was inspired from thermodynamics, then *information energy* is a concept inspired from the kinetic energy of gases. This concept was also used to asses poetry features related to poetic uncertainty as introduced and used for the first time by Onicescu [2] in the mid-1960s. Its definition resembles the definition of the kinetic energy as in the following

$$I = \frac{1}{2}p(a)^2 + \frac{1}{2}p(b)^2 + \cdots + \frac{1}{2}p(z)^2.$$

This statistical measure is also an analog of the moment of inertia from solid mechanics. This measures the *easiness* of rotation of the letters probability distribution $\{p(i)\}$ about the x-axis. Heuristically, the uniform distribution will have the lowest informational energy, as it has the lowest easiness of rotation about the x-axis, see Figure 2.

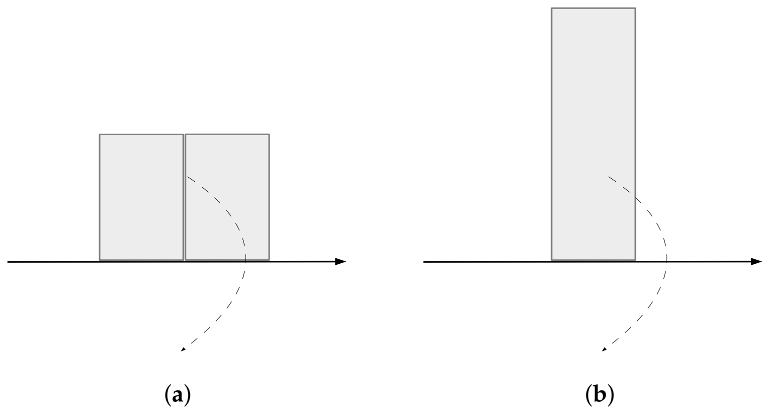


Figure 2. An interpretation of the informational energy as a momentum of inertia: The distribution (**a**) tends to rotate about the x-axis easier than distribution (**b**), so it has a smaller momentum of rotation.

This fact can be shown using Lagrange multipliers as follows. Let $p_i = p(i)$ denote the probability of each letter. We look for the distribution that minimizes $I = \frac{1}{2}\sum_{i=1}^{26} p_i^2$ subject to constraints $0 \leq p_i \leq 1$ and $\sum_{i=1}^{26} p_i = 1$. The associated Lagrangian is

$$L(p) = \frac{1}{2}\sum_{i=1}^{26} p_i^2 - \lambda(\sum_{i=1}^{26} p_i - 1).$$

Solving the Lagrange equations $\partial_{p_i} L(p) = p_i - \lambda = 0$, we obtain $p_i = \lambda = 1/26$, which is a uniform distribution. Since $\partial_{p_i}\partial_{p_j} L(p) = \delta_{ij}$, this solution corresponds to a global minimum.

Therefore, a short verse and a regular rhythm in poetry tend to increase the informational energy. On the other side, using a free verse tends to decrease the value of I. In addition to this, it has been noticed that entropy and informational energy have inverse variations, in the sense that any poem with a large information energy tends to have a smaller entropy, see Marcus [3]. We note that the entropy attains its maximum for the uniform distribution, while the informational energy reaches its minimum for the same distribution.

To get an idea about the magnitude and variation of the informational energy for the English language we provide the following table:

Title	Information Energy
Jefferson the Virginian	0.0332
Nietzsche	0.0334
Genesis	0.0348
Macbeth	0.0319
Romeo and Juliet	0.0323
Othello	0.0327
King Richard III	0.0314
Byron vol. I	0.0322
Byron vol. II	0.0324

For the sake of comparison with Romanian language we include the next table:

Author	Information Energy
Eminescu	0.0368
Ispirescu	0.0422
Creanga	0.0399

We note that English language has a smaller informational energy than Romanian, the range for the former being 0.031–0.032, while for the latter, 0.036–0.042.

It is worth noting that most entropy limitations encountered before carry also over to the informational energy.

The fact that it can exist a fine poem and a rubbish text with the same entropy and information energy (we can transform one into the other by just making some permutations of the letters in the poem) shows the drastic limitation of the usage of these two information tools. The main limitation consists of using probability of individual letters, which completely neglects the meaning casted in the words of the poem. In order to fix this problem, one should consider the random variable X to take values in a dictionary rather than in an alphabet. This way, the variable X, which is still finite, has thousands of outcomes rather that just 26. For instance, if in a poem, the word "luck" is used 5 times and the poem has 200 words, then we compute the probability $p(\text{luck}) = 5/200$. Doing this for all words, we can compute an informational entropy and informational energy at the words level, which makes more sense than at the letters level. It has been noticed that the words with the highest frequency in English are "the", "of", "end", "I", "or", "say", etc. Their empirical probabilities are $p(\text{the}) = 0.071$, $p(\text{of}) = 0.034$, etc. The interested reader can consult the words frequency graph given by Figure 1 of [4].

Consequently, if poem A uses each word equally often (fact that confers the poem a specific joyful structure) and poem B uses each word only once (fact that means the poem uses a richer vocabulary), then poem A will have the largest possible entropy and poem B will have the lowest. This fact is based on the fact that the equiprobable distribution reaches the highest entropy.

Unfortunately, even this approach of defining a hierarchical entropy at the level of words is flawed. If we consider two poems, the original one and then the same poem obtained by writing the words in reversed order, then both poems would have the same entropy and informational energy, even if the poetic sense is completely lost in the latter poem. This is because the order of words matter in conferring a specific meaning. The poetical sense is destroyed by changing the words order. This can be explained by considering each word as a cohesive group of letters with strong statistical inferences with nearby groups of letters. This problem has a fix using stochastic processes.

2.4. Marcov Processes

A mathematical model implementing the temporal structure of the poem is needed. The order of words can be modeled by a Markov process, by which each word is related with other words in the dictionary using a directional tree. Each node in this tree is associated with a word and each edge is assigned a transition probability describing the chance of going from one word to another.

The "markovian" feature refers to the fact that the process does not have any memory; it just triggers the next word, given one word, with a certain probability. For example, the word "magic" can be followed by any of the words "trick", "wand", "land", or "carpet". In a poem, using the phrase "magic trick" might not be very poetical, but using "magic wand" or "magic land" seems to be better.

From this point of view, writing a poem means to navigate through a directional tree whose nodes are words in a given vocabulary of a certain language, see Figure 3. Taking products of transition probabilities of edges joining consecutive words in the poem we obtain the total probability of the poem as a sequence of random variables (words) extracted from the dictionary. This approach involves a temporal structure between words, which marks its superiority from the aforementioned approaches. If one would have access to this directional tree and its probabilities then he/she would be able to generate new poems, statistically similar with the ones used to construct the tree.

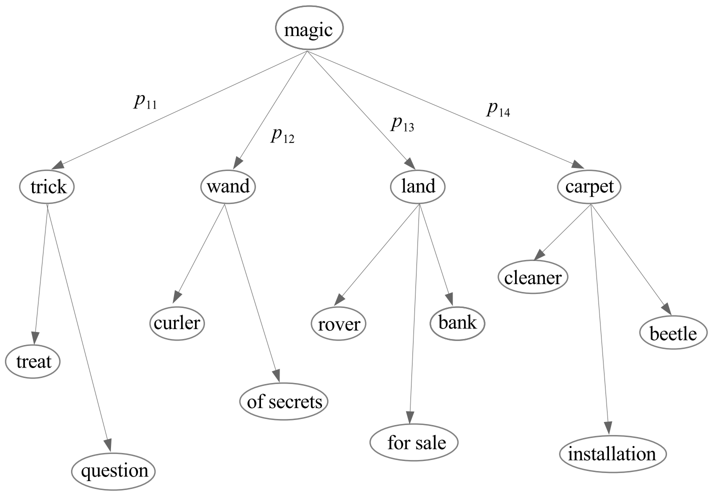


Figure 3. An example of directional tree of words in a dictionary.

This method in its current form is also limited. One reason is because the process does not have any memory. For instance, if a poem talks about "Odysseus" from the Greek mythology, then words as "legendary", "hero", "ingenious" will have a larger probability to appear later in the poem, even if they are not following directly the word "Odysseus".

A superior method, which is not markovian, will be described in the second part of the paper and regards the use of neural networks in poetry.

2.5. N-Gram Entropy

In his 1951 paper, Shannon [4] studied the predictability of the English language introducing the concept of the *N-gram entropy*. This is a sequence of conditional entropies, $F_0, F_1, F_2, \cdots$, given by

$$\begin{aligned} F_N &= -\sum_{i,j} p(B_i, j) \log_2 p(j|B_i) \\ &= -\sum_{i,j} p(B_i, j) \log_2 p(B_i, j) + \sum_{i} p(B_i) \log_2 p(B_i), \end{aligned}$$

where B_i is a block of $N-1$ letters, j is an actual letter, $p(B_i, j)$ is the probability of occurrence of the N-letter block (B_i, j), $p(B_i)$ is the probability of the block B_i and $p(j|B_i)$ is the conditional probability of letter j to follow the block B_i and is given by $p(j|B_i) = p(B_i, j)/p(B_i)$.

F_N can be considered as the conditional entropy of the next letter j when the preceding $N-1$ letters are given. Therefore, F_N measures the entropy due to statistics extending over N adjacent letters of the text. This approach is useful when one tries to answer the following question: *Given* $N-1$ *letters of a text, find the letter which is most likely to appear next in the text.*

The N-gram entropies, F_N, are considered as an approximating sequence of another object, a long-range statistic, called the *language entropy*, defined by

$$\mathbf{H} = \lim_{N\to\infty} F_N, \tag{1}$$

and measured in bits (per letter). Computing the language entropy of a given poetry author is a complex computational problem. However, the first few N-gram entropies can be computed relative easily and were found by Shannon [4].

First, regardless of the author, $F_0 = \log_2 26 = 4.7$ (bits per letter), given that the language is English (with an alphabet of 26 letters). The 1-gram involves the entropy associated with letter frequency

$$F_1 = -\sum_{j=1}^{26} p(j)\log_2(p(j)) = 4.14,$$

for English language. For any poetry author, his/hers F_1 entropy should be smaller than 4.14 (bits per letter) due to the style structure constraints. It is worth noting that $F_1 = H$, where H is the aforementioned informational entropy. The interpretation of F_1 is the number of random guesses one would take on average to rich the correct next letter in a text, without having any knowledge about previous letters.

The digram F_2 measures the entropy over pairs of two consecutive letters as

$$\begin{aligned} F_2 &= -\sum_{i,j=1}^{26} p(i,j)\log_2(j|i) \\ &= -\sum_{i,j=1}^{26} p(i,j)\log_2 p(i,j) - F_1 = 3.56 \text{ bits.} \end{aligned}$$

F_2 can be interpreted as the number of random guesses one would take on average to rich the correct next letter in a text, knowing the previous letter.

The trigram F_3 can be also be computed as

$$\begin{aligned} F_3 &= -\sum_{i,j,k=1}^{26} p(i,j,k)\log_2(k|i,j) \\ &= -\sum_{i,j,k=1}^{26} p(i,j,k)\log_2 p(i,j,k) - F_2 = 3.3 \text{ bits.} \end{aligned}$$

Similarly, F_3 can be interpreted as the number of random guesses one would take on average to rich the correct next letter in a text, knowing the previous two letters.

2.6. Language Entropy

In the following, we shall provide an approximative formula for computing the language entropy (1). If B is a letter block of length N we shall use notation $|B| = N$ to denote the length of the block B. We define the entropies sequence

$$H_N = -\sum_{|B|=N} p(B) \log_2 p(B). \tag{2}$$

This sum is taken over 26^N terms, most of which being equal to zero (when the block B does not appear in the text, the term $p(B)\log_2 p(B)$ is 0). For instance, $H_1 = -\sum_j p(j)\log_2 p(j) = H = F_1$ and $H_2 = -\sum_j p(i,j)\log_2 p(i,j)$. Now, F_N can be written recursively in terms of H_N and H_{N-1} as

$$\begin{aligned} F_N &= -\sum_{i,j} p(B_i, j)\log_2 p(B_i, j) + \sum_i p(B_i)\log_2 p(B_i) \\ &= H_N - H_{N-1}. \end{aligned}$$

Iterating, we obtain $F_1 = H_1$, $F_2 = H_2 - H_1$, $F_3 = H_3 - H_2, \cdots, F_N = H_N - H_{N-1}$. Then adding and performing all cancellations, yields

$$H_N = F_1 + F_2 + \cdots + F_N.$$

If the sequence $(F_N)_N$ is convergent to the language entropy $\mathbf{H}$, as $N \to \infty$, then a well known property states that the sequence of arithmetic averages also converges to the same limit,

$$\lim_{N\to\infty} \frac{F_1 + F_2 + \cdots + F_N}{N} = \mathbf{H}.$$

This leads to the following formula for the language entropy

$$\mathbf{H} = \lim_{N\to\infty} \frac{H_N}{N}. \tag{3}$$

In other words, H_N tends to increase linearly with respect to N, and the long run proportionality factor is the language entropy.

2.7. Conveyed Information

Shannon's work [4] is mainly concerned with the predictability of the English language. One way to continue this idea is to ask the following question: *How much information is conveyed by a block of $N - 1$ letters about the next letter in the text?* This is related to the previous question of finding the conditional probability of the next letter given a previous letter block. For this we recall the following notion of information theory, which can be found in more detail in [5].

Let X and Y be two random variables. The information conveyed about X by Y is defined by $I(X|Y) = H(X) - H(X|Y)$, where $H(X)$ is the entropy of random variable X and $H(X|Y)$ is the conditional entropy of X given Y. In our case X represents the random variable taking values in the alphabet $\{a, b, \cdots, z\}$, while Y is the random variable taking $(N-1)$-letter blocks values; each of these blocks can be considered as a function $f : \{1, 2, \cdots, N-1\} \to \{a, b, \cdots, z\}$.

The information conveyed by an $(N-1)$-letter group about the next letter in the text will be denoted by $I_{1,N-1}$ and is computed as follows

$$\begin{aligned}
I_{1,N-1} &= I(X|Y) = -\sum_i p(i)\log_2 p(i) + \sum_{i,j} p(B_i,j)\log_2 p(j|B_i) \\
&= H_1 - F_N = H_1 - (H_N - H_{N-1}) \\
&= H_1 + H_{N-1} - H_N.
\end{aligned}$$

Similarly, we can define the information conveyed by a $(N-2)$-letter group about the next 2-letter group in the text by

$$\begin{aligned}
I_{2,N-2} &= -\sum_{i,j} p(i,j)\log_2 p(i,j) + \sum_{i,j,k} p(B_i,j,k)\log_2 p(j,k|B_i) \\
&= H_2 - H_N + H_{N-2} \\
&= F_1 + F_2 - F_{N-1} - F_N.
\end{aligned}$$

This leads inductively to the formula of the information conveyed by an M-letter group about the next k-letter group in a text as

$$I(k,M) = H_k + H_M - H_{M+k}.$$

We have the following information inequalities:

Proposition 1. *For any integers $k, M \geq 1$ we have:*

$$(i) \qquad H_k + H_{M+k} \geq H_M$$

$$(ii) \qquad \sum_{j=1}^{k} F_j \geq \sum_{j=1}^{k} F_{M+j}.$$

Proof. (i) Since the conveyed information is non-negative, $I(X|Y) \geq 0$, then $I(k,M) = H_k + H_M - H_{M+k} \geq 0$, fact that implies the first inequality.

(ii) The previous inequality can be written telescopically as

$$(F_1 + \cdots + F_k) + (F_1 + \cdots + F_M + \cdots + F_{M+k}) \geq F_1 + \cdots + F_M,$$

which after cancelling similar terms leads to the desired inequality. □

It is worth noting that for $k = 1$, the second inequality becomes $F_1 \geq F_{M+1}$, for any $M \geq 1$, which implies $F_1 \geq F_j$, for all $j \geq 1$. This shows that the entropy associated with letter frequency, F_1, is the largest among all N-gram entropies. This fact can be also noticed from the numerical values presented in Section 2.5.

Another remark is that inequalities of Proposition 1 are strict as long as the text bears some meaningful information. This follows from the fact that $I(X|Y) = 0$ if and only if X and Y are independent random variables.

2.8. N-Gram Informational Energy

Inspired by the work of Shannon [4] and Onicescu [2], we introduce next the concept of *N-gram information energy*, which is a statistic defined by

$$I_N = \frac{1}{2}\sum_{|B|=N} p^2(B), \tag{4}$$

where $B = (b_{i_1}, \cdots, b_{i_N})$ is a letter block of length N and $p(B)$ is the probability that the block B appears in the text. We note that $I_1 = \frac{1}{2}\sum_{j=1}^{26} p^2(j)$ is the letter information energy introduced before. Inspired by formula (3), we define the *language information energy* by

$$\mathbf{I} = \lim_{N\to\infty} \frac{I_N}{N}.$$

The existence of this limit constitutes a future direction of research and won't be approached in this paper.

2.9. Comparison of Two Texts

In this section, we shall introduce a method of comparing two texts, by evaluating the information proximity of one text with respect to another. We shall first consider a benchmark text in plain English with letter distribution p. This text will provide the distribution of letters in English.

The second text can be a poetry text, also in English, with letter distribution q. We shall compare the texts by computing the Kullback–Leibler relative entropy, $D(p||q)$. This measures a certain proximity between distributions p and q as in the following sense. If the benchmark text is considered efficient in expressing some idea, using poetry to express the same idea can be less efficient; thus, the expression $D(p||q)$ measures the inefficiency of expressing an idea using poetry rather than using plain English. From the technical point of view, the Kullback–Leibler relative entropy is given by the formula

$$D(p||q) = \sum_{i=1}^{26} p(i) \log_2 \frac{p(i)}{q(i)}.$$

The text with letter distribution q will be called the *test text*, while the one with distribution p, the *benchmark text*. We shall consider the benchmark text to be the entire text of the book *Jefferson the Virginian* by Dumas Malone, as Shannon himself considered it initially, while investigating the prediction of English language, [4]. The first test text is the first volume of Byron poetry. A simulation in 'R' provides the relative entropy value

$$D(\text{Dumas}||\text{Byron}) = 0.01368 \text{ bits}.$$

The next test text is taken to be *Othello* by Shakespeare, case in which

$$D(\text{Dumas}||\text{Shakespeare}) = 0.02136 \text{ bits}.$$

This shows a larger measure of inefficiency when trying to express Dumas's ideas using Shakespearian language rather than Byron-type language. This can be explained by the fact that Byron used a variant of English closer to Dumas' language rather than Shakespeare. Consequently, the language used in *Othello* is farther from English than the one used in Byron's poetry.

Considering further the test text as the *Genesis* chapter, we obtain

$$D(\text{Dumas}||\text{Genesis}) = 0.03671\text{bits}.$$

This value, larger than the previous two, can be explained by the even older version of English used by the Bible.

These comparisons are based solely on the single letter distribution. It is interesting to consider a future direction of research, which compares two texts using a more complex relative entropy defined by

$$D_N(p||q) = \sum_{|B|=N} p(B) \log_2 \frac{p(B)}{q(B)},$$

which is a Kullback–Leibler relative entropy considering N-letter blocks rather than single letters. We obviously have $D_1(p||q) = D(p||q)$.

One question is whether the quotient $\frac{1}{N} D_N(p||q)$ approaches a certain value for N large. In this case, the value of the limit would be the *languages relative entropy*.

Another variant of comparing two texts is to compute the cross-entropy of one with respect to the other. The cross-entropy formula is given by

$$CE(p, q) = -\sum_i p(i) \log_2 q(i).$$

Since an algebraic computation provides

$$CE(p, q) = D_{KL}(p||q) + H(p),$$

then the cross-entropy between two texts is obtained by adding the entropy of the benchmark text to the Kullback–Leibler relative entropy of the texts. Since $D_{KL}(p||q) \geq 0$, then $CE(p, q) \geq H(p)$. Given the distribution p, the minimum of the cross-entropy $CE(p, q)$ is realized for $q = p$, and in this case $CE(p, q) = H(p)$.

The idea of comparing two texts using the relative or cross-entropy will be used to measure the information deviation of RNN-generated poems from genuine Byron poetry.

2.10. Neural Networks in Poetry

The comprehension of hidden probabilistic structures built among the words of a poem and their relation with the poetical meaning is a complex problem for the human mind and cannot be caught into a simple statistic or mathematical model. This difficulty comes from the fact that each poet has his/her own specific style features, while using an enormous knowledge and control over the statistics of the language he/she writes in. Therefore, we need a system that could be able to learn abstract hierarchical representations of the poetical connotations represented by words and phrases.

Given the complexity of this task, one approach is to employ machine learning. This means to look for computer implemented techniques that are able to capture statistical regularities in a written text. After learning the author's specific writing patterns, the machine should be able to adapt to the author's style and generate new texts, which are statistically equivalent with the ones the machine was trained on.

2.10.1. RNNs

It turned out recently that one way to successfully approach this complex problem is to employ a specific type of machine learning, called Recurrent Neural Network (RNN), which date back to 1986 and is based on David Rumelhart's work [6]. This belongs to a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence, similar to the directional tree previously mentioned in Section 2.4.

The main feature of these networks is allowing the network to exhibit temporal dynamic behavior. The network input and the output vectors at time t are denoted respectively by X_t and Y_t, while h_t represents the hidden state at time t. In its simplest case, the transition equations of an RNN take the following form

$$\begin{aligned} h_t &= \tanh(W h_{t-1} + U X_t + b) \\ Y_t &= V h_t + c, \end{aligned}$$

see Figure 4. Thus, the hidden states take values between -1 and 1, while the output is an affine function of the current hidden state. Matrices W and U represent the hidden-to-hidden state transition and input-to-hidden state transition, respectively. b and c denote bias vectors. During training the network parameters (W, U, b, c) adjust such that the output sequence $(Y_t)_t$ gets as close as possible to the sequence of target vectors, $(Z_t)_t$. This tuning process, by which a cost function, such as

$$C(W, U, b, c) = \frac{1}{2}\sum_{t=1}^{T} \|Z_t - Y_t\|^2$$

is minimized, is called *learning*. This idea is the basis of many revolutionizing technologies, such as: machine translation, text-to-speech synthesis, speech recognition, automatic image captioning, etc. For instance, in machine translation from English to French, $X_1, X_2, \cdots X_T$ represent a sentence in English, and $Y_1, \cdots, Y_N$ represents the French translation sentence. The statistical relationships between English and French languages is stored in the hidden states and parameters of the network, which are tuned in such a way that the translation is as good as possible.

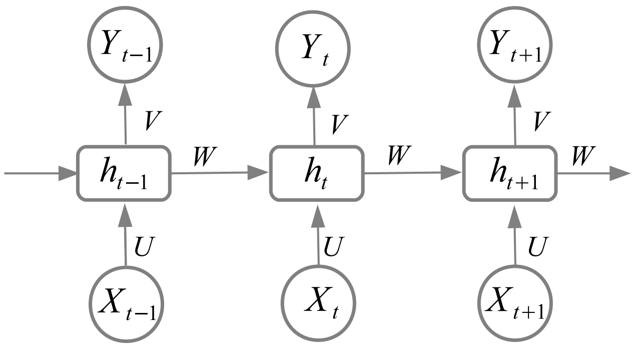


Figure 4. A simple RNN, with one layer of hidden units.

Each network bares a certain *capacity*, which can be interpreted as the network ability to fit a large variety of target functions. This depends on the number of its parameters. If the network contains just a few units, the number of parameters is small and its capacity is low. Consequently, the network will underfit the data; for instance, considering the case of the previous example, the network cannot translate complex sentences. For this reason, the capacity has to be enhanced. There are two ways to do that: (i) by increasing the number of hidden units (horizontal expansion) and (ii) considering several layers of hidden states (deep neural network).

When increasing the network capacity two difficulties can occur: the vanishing gradient problem and the exploding gradient problem. The first problem can be addressed by considering special hidden units such as LSTM cells, which were introduced in 1997 by Hochreiter and Schmidhuber [7]. We won't get into the details here, but the reader is referred to consult [5].

In the rest of the paper we shall use an RNN consisting of a single-layer of LSTM cells to learn statistical features of Byron's poems and then use these parameters to generate new Byron-type poems. The way we shall generate new poems is to input a certain "seed" phrase and use the trained network to generate the prediction of the next 400 characters which would follow the provided seed. We shall obtain a sequence of "poems", as training progresses, of increasing quality. We shall then asses their informational deviation from a genuine Byron text, using the statistical measures introduced in the first part of the paper. When this difference of the similarity is small enough we shall stop training.

The next sections take the reader through all the steps necessary for the aforementioned goal. These include data collection, data processing and cleaning, choosing the neural model, training, and conclusions.

2.10.2. Data Collection

Any machine learning model is based on learning from data. Therefore, the first step is to collect data, as much and as qualitative as possible. Regarding our project, I found Byron's works online under Project Gutenberg at www.gutenberg.org. Since the website states "this eBook is for the use of anyone anywhere at no cost and with almost no restrictions whatsoever", it is clear we may use it. Lord Byron's work counts seven volumes. All contain poetry, but the second volume, which deals with letters and journals. Among all these files, we shall use for training only volume I of poetry, which can be found at http://www.gutenberg.org/ebooks/8861.

The reason for which we had restricted our study to only one volume is the limited processing power available, as the training was done on a regular PC-laptop.

2.10.3. Data Processing and Clean-Up

The quality of data is an important ingredient in data preparation step, since if data are not well cleaned, regardless of the neural architecture complexity used, the results won't be qualitative good.

In our case, almost all poems contain lots of footnotes and other explanatory notes, which are written in plain English by the publisher. Therefore, our next task was to remove these notes, including all forewords or comments from the publisher, keeping only pure Byron's work. This might take a while, as it is done semi-manually. The reader can find the cleaned-up text file at http://machinelearningofannarbor.org/book1.txt.

2.10.4. Choosing the Model

The neural network model employed in the analysis of Byron's poetry is presented in Figure 5. It consists of an RNN with 70 LSTM cells (with hidden states h_i, $1 \leq i \leq 70$), on the top of which is added a dense layer with 50 cells. We chose only one layer in the RNN because if considering a multiple layer RNN the number of network parameters increases, which might lead eventually to an overfit of data, which is restricted at this time to only one volume of poetry. The output of the network uses a softmax activation function, which provides a distribution of probabilities of the next predicted character. The input is done through 70 input variables, $X_1, \cdots, X_{70}$. If the poetry text is written as a sequence of characters (not necessary only letters) as

$$i_1, i_2, i_3, i_4, i_5, \cdots, i_{70}, i_{71}, i_{72}, i_{73}, \cdots, i_N,$$

then the text was parsed into chunks of 70 characters and fed into the network as follows. The first input values are the first 70 characters in the text as

$$(x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, \cdots, x_{70}^{(1)}) = (i_1, i_2, i_3, \cdots, i_{70}).$$

The next input sequence is obtained by shifting to the right by a step of 3 characters as

$$(x_1^{(2)}, x_2^{(2)}, x_3^{(2)}, \cdots, x_{70}^{(2)}) = (i_4, i_5, i_6, \cdots, i_{73}).$$

The third input sequence is obtained after a new shift as

$$(x_1^{(3)}, x_2^{(3)}, x_3^{(3)}, \cdots, x_{70}^{(3)}) = (i_7, i_8, i_9, \cdots, i_{76}).$$

Since the text length is $N = 374{,}363$ characters, then the number of input sequences is 124,765. It also turns out that the number of characters in Byron's poems is 63. This includes besides 26 letters and 10 usual digits, also other characters such as coma, period, colon, semicolon, quotation marks, question marks, accents, etc., see Figure 6. Therefore, the network output will be 63-dimensional.

The activation function in the dense layer is a ReLU function (which is preferable to sigmoid activation function as it usually provides better accuracy). The learning is using the Adam minimization algorithm—an adaptive momentum algorithm, which is a variation of the gradient descent method, with adjustable learning rate. Training occurs using batches of size 80 at a time and it trains for 150 epochs. Training takes roughly 3 min per epoch, taking about 7 h for all 150 epochs.

Since this is a multi-class classification problem (with 63 classes), the loss function is taken to be the categorical cross-entropy. Each class corresponds to a character given in Figure 6.

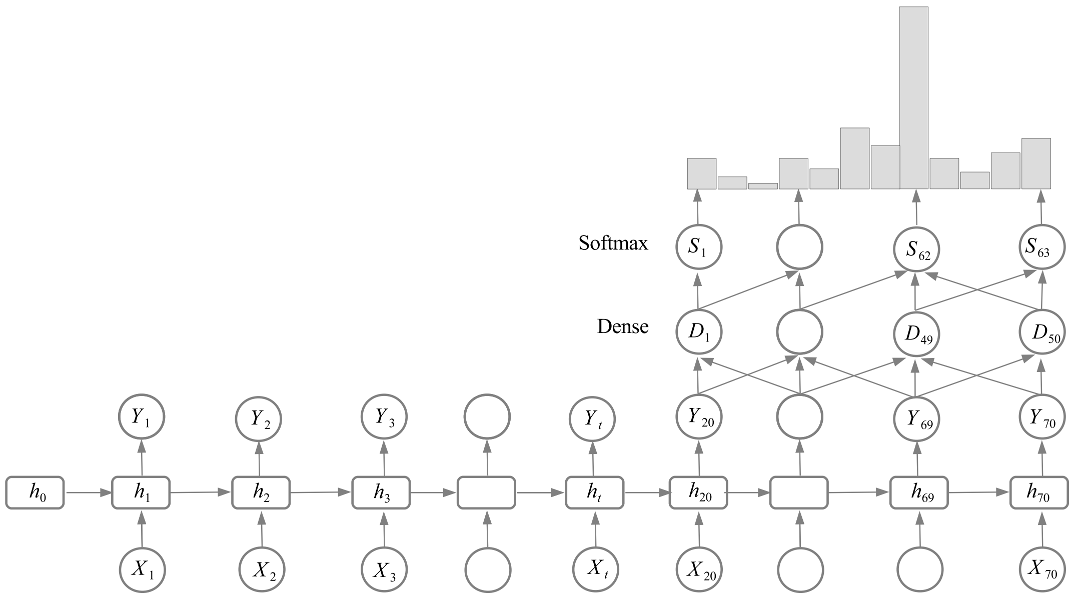


Figure 5. A simple RNN with 70-LSTM cells, with an overlap of one dense layer and a softmax activation output.

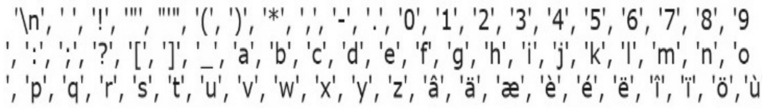

'\n', ' ', '!', '"', '"', '(', ')', '*', ',', '-', '.', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9
', ':', ';', '?', '[', ']', '_', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o
', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', 'â', 'ä', 'æ', 'è', 'é', 'ë', 'î', 'ï', 'ö', 'ù

Figure 6. The characters present in Byron's poetry. Each character is included between single quotation marks.

If q denotes the probability density of the next character forecasted by the network and p is the probability density of the true character in the text, then during training the network minimizes the cross-entropy

$$CE = -\sum_{i=1}^{63} p(i) \ln q(i).$$

Therefore, learning tries to minimize the average number of bits needed to identify a character correctly using the probability distribution q, rather than the true distribution p. Since the text contains poetry from only the first volume, in order to avoid overfitting, a dropout of 4% has been introduced after both the LSTM and the dense layers.

The program was implemented in Keras and can be found for the sake of reproducibility at the page http://machinelearningofannarbor.org/Byron_Keras.txt.

3. Results and Discussion

During training, we had monitored the loss function expressed by the cross-entropy, which decreases steadily from 3.5 to 1.37 over 150 epochs, see Figure 7. This decrease in loss is a sign of good learning. We have tried different hyperparameters (batch size, learning rate, number of epochs, step, parsing length, number of LSTM cells) and arrived to the ones used in this paper as being pretty close to optimal. The generated text seems to resemble more and more the genuine Byron poetry as the training process progresses and the loss decreases. Theoretically, if the cross-entropy loss is zero, $CE(p, q) = 0$, then $q = p$, namely the generated text has the same letter distributions as the genuine Byron text.

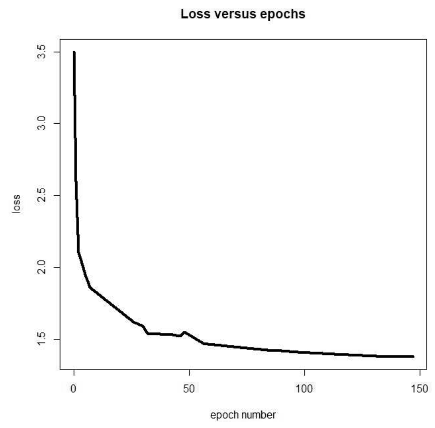


Figure 7. The cross-entropy loss decreases as the number of epochs increases.

In the beginning, after the first epoch, the generated text looks more like a random sequence of groups of letters. There are neither English words nor poetry structure present after the beginning seed phrase:

what! not a word!–and am i the the cisseth pantses susthe the the lof lelen ban depilers in the shele whe whas the as hof bathe the with lens on bo sint the wor thire ched the the the ghordes ang the aned fors and rore dis the the the sors,

The loss after the first epoch is 2.605. The entropy of the generated text is $H(1) = 3.694$.

After the second epoch the loss decreased to 2.187 and the text entropy increased to $H(2) = 4.043$. The network already learnt some verse structure, even if most words are still not in English:

Wake a wildly thrilling measure.
There will my gentle girl and i,
The bare pus the soun mare shes and the stist,
And the and will sost an fing sote no on ass som yian,
A whis for atile and or souns and pey a porin dave sis bund and are.

Tha thaigh cill list the the beraich be for withe hond,
Far om dill and butting of wole,
O sould af mare hor wore sour;
To will the ast dore he ares ereis s soar?

After epoch 26 the cross-entropy loss had a substantial decrease to 1.62, while the entropy is $H(26) = 3.99$. In this case the generated text looks a lot more like English:

No just applause her honoured name
Of idiots that infest her age;
No just applause her honoured name of strain,
And still the last his breathed lings his press,
Thou not his art of heart to loves of may,
The spare that mays the ding the rain,
Which cares the shep her loves of prose;
Then heart the such the gleam the live decear;
This still the gard, our thee in parter's fiest
The love through may the word,
When bay and mights of some the song the new
The sits again, and sing...

After 30 epochs, the loss is 1.59, while the entropy is $H(30) = 4.057$ (getting closer to Byron's genuine poetry entropy of 4.184); the outcome has an improved English:

Outhful eclogues of our pope?
Yet his and philips' faults, of differs still;
To hours, where still the promenty glow.
The dearts and the tome to dark,
And to the child and fears of the and the farre;
Though her decerance more thy poes's to dark,
And path of a still howard of day!
Your conful in so later her stord,
And praise of the dear the day,
In proded the former frem this such,
In the soul in sumery vain,
And so and the Lamble...

In the following, we shall provide the epoch number, loss value and entropy and the generated texts for a few more samples. We note the decrease in loss and increase in entropy values:

Epoch 43, loss 1.54, entropy $H(43) = 4.01$:

The hall of my fathers, art gone to decay;
In thy once smiling garms and stranger striating and and his die;
The bounding poud the spreise of fall the live,
The classic poed to the dear of dray–
To must the dear a strong the combon the forth
The great by the dear with this the breathing, well,
In all the percid the thanding showe;
When prate the that a some his will the say;
Thou the eyes of the bard, and thought the sungod a fill;

Epoch 56, loss 1.4691, entropy $H(56) = 3.98$:

A transport of young indignation,
With fervent contempt evermore to disdain you:
I seembrate to the sought for the faint shill
The gark her sweets thou greated of the sain,
A dele the praise the seem and merben meet,
I change eye may of thy spirit lead,
No verse thy fain as gale the sight.

5.
Then, the more the durming fort confine!
The pallage in once the piling fain,
And not stranger the saint despise me to fair,
The lovering deep and the prines of critic ...

Epoch 75, loss 1.4370, $H(75) = 4.07$:

Few short years will shower
The gift of riches, and the pride of power;
E'en now a name the seems who the prophears to fear,
And stripling band the heart they discounts all the trays!
One back with must a seem shill the pands,
The love and the forger disgreand ford;

And fur the versance the lead at lade;
Who bard my path, and bounce love alone;
The daughter land admartuner sweet the speak, while,
May speak and the some fain the gold my pease,...

Epoch 90, loss 1.4187, entropy $H(90) = 3.94$:

And so perhaps you'll say of me,
In which your readers may agree.
Still I write on, and so mine thee;
Yet with the heart of earmoness of and cartale.

1.
What and a seem the only soul its me,
Who balling all of the shall fordst these page bare:
And the shall helf the partless deep for the cheep;
What shine! though all some can must here the seif.
The tells the palling, while the charms the plies;
And for the sight! when they the heart the prine,

Epoch 113, loss 1.3955, entropy $H(113) = 3.99$:

Hope's endeavour,
Or friendship's tears, pride rush'd between,
And blotted out the mander great.
Who fate, though for the right the still grew,
To siek'st envesing in stream and tome,
In parth will be my faster and dear,
To gromend the dare, which lead the glord,
The pence with the steel to the foem
And the tarth, i change the rofty cound,
This cale the ang repire the view,
For the the saight we fore before,
And the fall to the deastex flow,...

Epoch 138, loss 1.3847, entropy $H(138) = 3.93$:

Live me again a faithful few,
In years and feelings still the same,
And I will fly the hoother shine,
And child the dast the tarth with
The tone the fall thy stain'd or sweet,
And then long the pang the nommer reat,
When lut the well the sames like the stand,
When wit the changuid our foight with still thee,
What mingtred her song each senate,
And brand the senty art steps record,
The spring the first by poesion the faint,
Some light the blest the rest to sig...

Epoch 146, loss 1.377, entropy $H(146) = 4.07$:

Would have woman from paradise driven;
Instead of his houris, a flimsy pretence,

While voice is and the buld they consuce,
Which menting parragues, chilfoued the resure
And to might with from the gold a boy,
The seems my son to cartan's plaintus for the
The ear in the still be the praces,
And life the disonge to shere a chance;
The mowting pours, the sonite of the fore
The pures of sain, and all her all,
And the presert, conal all, or the steems,...

Epoch 148, loss 1.37, entropy $H(148) = 4.05$:

Lengthened line majestic swim,
The last display the free unfettered limb!
Those for him as here a gay and speet,
Revile to prayial though thy soul and the such
Roprowans his view where the happling hall:
He feelin sense with the gloolish fair to the daw;
The scane the pands unpressar of all the shakenome from.

As it can be inferred from all presented samples, even if the generated Byron poetry is satisfactory, it is not perfect. There are words made up by the network, such as "happling hall" or "unpressar", which cannot be found in any English dictionary. However, for one who does not know the language well, they look quite English. The way the network invents this words is using the statistical relation between letters, without understanding their meaning. Therefore, a human is using words by their meaning, while a machine is using words by their statistical inference from letters distribution.

4. Conclusions and Future Directions

There are a few directions of improvement that we shall discuss next. One simple direction is to use an enhanced training data, including all volumes of Byron's work. This way, the network will have more information to learn from. Another direction of improvement, which can be used simultaneous with the former, is to use a deep neural network. A deeper learning can be accomplished if more than one layer of LSTM cells are included in the RNN network. This more complex architecture allows for learning more nonlinear relations between letters and words, tapping even into the poetry meaning. Additionally, a longer chain of LSTM cells on each layer would increase the efficiency of learning, at the price of computational expense. To overcome this problem a GPU computing unit (or a cluster of GPUs) has to replace the usual laptop CPU unit. We believe the increase in the computational power will soon be able to generate texts which are indistinguishable from genuine Byron poetry.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Shannon, C. A mathematical theory of communication. *Bell. Syst. Tech. J.* **1948**, *27*, 379–423
2. Onicescu, O. Energie informationala. *St. Cerc. Math.* **1966**, *18*, 1419–1430.
3. Marcus, S. *Poetica Matematica*; Editura Academiei Republicii Socialiste Romania: Bucharest, Romania, 1970.
4. Shannon, C. Prediction and entropy of printed English. *Bell Syst. Techn. J.* **1951**, *30*, 50–64. [CrossRef]
5. Calin, O. *Deep Learning Arhitectures—A Mathematical Approach*; Springer Series in Data Sciences; Springer: New York, NY, USA, 2020.

6. Williams, R.J.; Hinton, G.E.; Rumelhart, D.E. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536.
7. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]