Does it actually work? An analysis of the Expected Value Equation of Heaps' law.

By: Pierre Aucoin

Project type: Mad Data Scientist

Background and Thesis Statement:

Many data scientists have tried to use statistics to analyze texts. As part of this research, there have been many formulas created to help in this endeavour, specifically Heaps' law, which is a mathematical formula that shows a relationship between the total number of words in a text and the overall size of the vocabulary (which is defined as the number of unique words). However, I am interested in seeing if what was found in this research works with actual texts, as there is a difference between theoretical and practical success when it comes to applying formulas such as Heaps' law. In this project, I will be researching the expected value of Heaps' law and seeing if it is statistically accurate to what is found in actual texts.

Hypothesis:

I hypothesize that the expected value equation for Heaps' law will not be accurate for texts. This is because, in preliminary readings regarding this topic, I have read that there are statistically significant differences at times between the actual size of the vocabulary and the expected size of the vocabulary.

Data and Methodology:

For this project, I will use 250 PubMed abstracts, 250 abstracts created from GPT-Neo with 1.3 billion parameters, 250 abstracts created from GPT-Neo with 125 million parameters and 250 abstracts created from GPT-Neo with 2.7 billion parameters and determine whether the expected value equation is accurate for each set of data.
These have already been put into a .pkl file to be used in R code. Here is the link to the data: https://huggingface.co/datasets/rachel6603/PubMed-Mimic-Gpt-Neo.

In this project, I will only use the first 250 abstracts from each section.
I will use the expected value formula laid out in the Chacoma and Zanette paper: "*Heaps' Law and Heaps' functions in tagged texts: evidence of their linguistic relevance"* and apply it to the 250 abstracts and see how the formula expects the vocabulary size to change as more abstracts are being observed (https://royalsocietypublishing.org/doi/full/10.1098/rsos.200008).

Once completed, I should be able to tell whether the expected value formula is statistically accurate to the actual size of the vocabulary. To determine whether the expected value is significantly accurate, I will also look at the Variance formula outlined in the Chacoma and Zanette paper. If the expected value formula highlighted in the Chacoma and Zanette paper is significantly accurate, then the actual size of the vocabulary should fall within two standard deviations, which is just the square root of the variance equation above or below the expected value at every point. If the actual size of the vocabulary falls outside of this range, then the equation for the expected value is not significantly accurate.

Formula Explanation:

When working with Heaps' law, there is a formula to determine the expected vocabulary size for a document of n words. This equation is:

$E[V_n]=V-\sum_{i=1}^{V}\frac{\binom{N-N_i}{n}}{\binom{N}{n}}$

V is the vocabulary size, N is the total number of tokens (words), n is the number of words in the sample, and N_i is the number of times a specific token appears in the population. Something that I noticed is that the second part of the expected value equation is a hypergeometric distribution, which is in the form:

$P(x)=\binom{m}{x}\frac{\binom{n}{k}}{\binom{m+n}{k}}$

In this situation, x is 0, m is Ni, n is N-Ni, and k is n, which creates the second part of the expected value equation.

For this project, to determine whether this equation accurately represents the data, I also need to find the variance equation for Heaps' law so I know how far the actual values can be from the expected value and still have the equation work well. For Heaps' law, the Variance Equation is:

$V[V_n]=\sum_{i=1}^{V}\frac{\binom{N-N_i}{n}}{\binom{N}{n}}*\left(1-\frac{\binom{N-N_i}{n}}{\binom{N}{n}}\right)+2\sum_{r=2}^{V}\sum_{s=1}^{r-1}\frac{\binom{N-(N_r+N_s)}{n}}{\binom{N}{n}}-\frac{\binom{N-N_s}{n}}{\binom{N}{n}}*\frac{\binom{N-N_r}{n}}{\binom{N}{n}}$

N_r and N_s are the number of times a specific token appears in the population. This equation is very demanding on R as the second half of the equation has (V choose 2) components, as we get all combinations of two different tokens in the vocabulary set.

Also, because both equations heavily rely on the hypergeometric formula, they can be approximated through binomial approximation, which means that we have two other potential equations, these being:

$E[V_n]=V-\sum_{i=1}^{V}(1-\frac{N_i}{N})^n$

$V[V_n]=\sum_{i=1}^{V}(1-\frac{N_i}{N})^n(1-(1-\frac{N_i}{N})^n)+2*\sum_{r=2}^{V}\sum_{s=1}^{r-1}(1-\frac{N_r+N_s}{N})^n-(1-\frac{N_r}{N})^n(1-\frac{N_s}{N})^n$

These equations are made from the assumption that (N_i/N) approaches a value p as N approaches infinity, which allowed me to convert the hypergeometric distribution to a binomial one.

Procedure and Analysis:

For each graph created, the red line is the expected value while the red dotted lines are 2 standard deviations above and below that. The blue line is the approximated expected values using binomial approximation, while the dotted lines are 2 standard deviations above and below that. Finally, the orange line is the actual size of the vocabulary.

Before graphing the data set, it is important to clean the abstracts. This removes items such as punctuation, numbers, and other terms so that no tokens are double-counted for being unique. For example, if I did not clean this sentence and I put this as data, the word data would be split into two different vocabulary words (data, and data).

Now that the data is clean, I can now get the values of the variables needed for the equations. Using the R code, I can get all the variables needed for the two equations.

Firstly, I calculated the expected values and variance for the PubMed data. I am using R functions to help calculate all the expected values and variances using both the hypergeometric and binomial versions. In the graph that has the actual data, both versions of the expected value equation and two standard deviations above and below the two expected values, I noticed that when looking at the values comparing the expected value and the binomial approximation for the expected value, they are not similar at all, as the two sets of lines do not stay close together as the number of words increases. When looking at the actual vocabulary size against the number of words compared to the expected values, we see that the two lines seem to match up precisely at (N,V). This makes sense as when looking at the equation for the Expected Value, if N=n, then the expected value will be V. However, there is an issue, and that is before the number of words gets to N, the actual size of the vocabulary stays significantly outside of the range of two standard deviations above or below the expected value (the dotted red lines on the graph). This shows that without using the entire data set as a sample, the expected value equation does not accurately predict the size of the vocabulary of a data set, given the number of words in it.

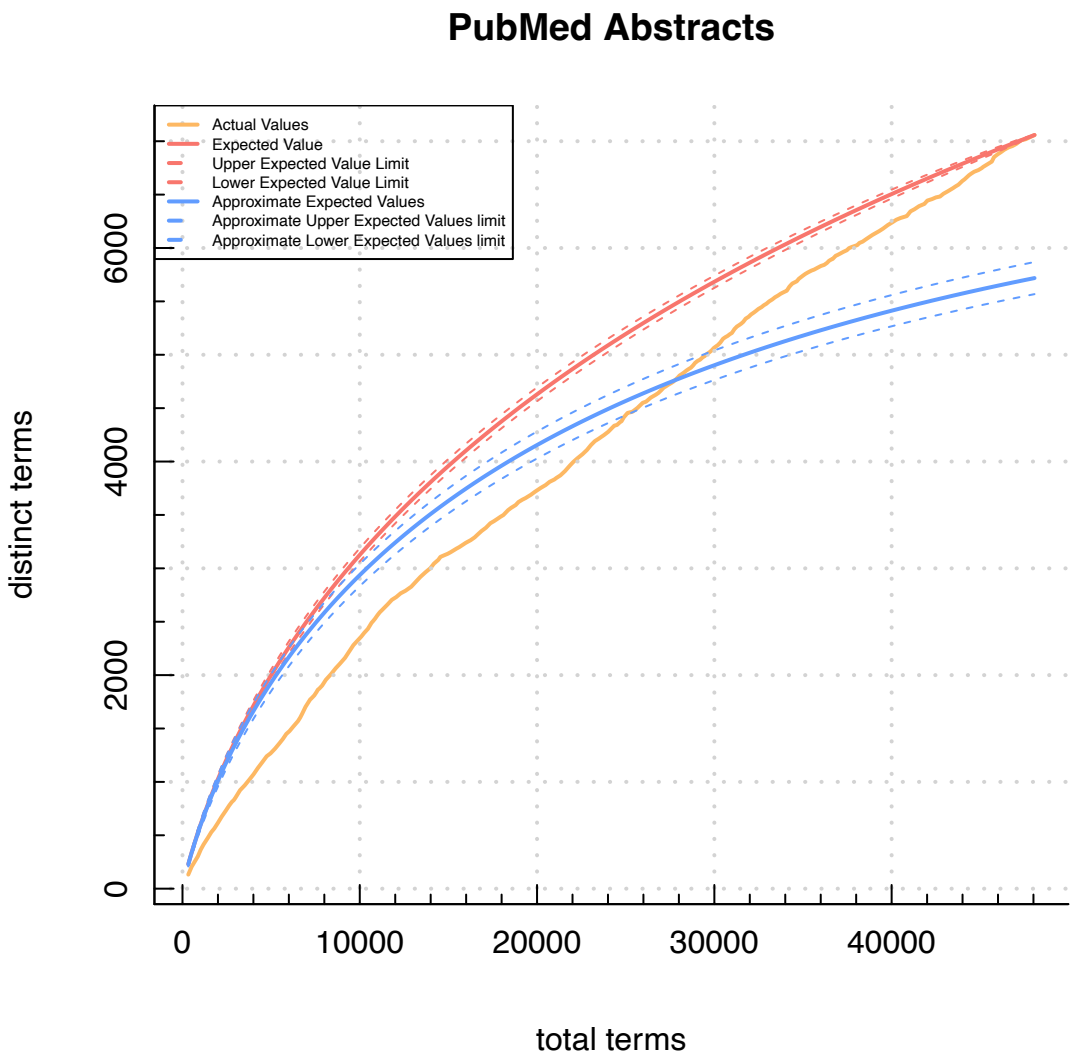

Figure 1: Graph of the expected value equations and the actual data for 250 PubMed abstracts.

Next, I did the same thing with abstracts created by GPT-Neo with 125 million parameters. This graph created similar results to those of the PubMed abstracts, although there were slight differences. Both the hypergeometric and binomial versions of the expected value and variance equations provide similar values when the total number of terms is small. As the number of words approaches N, the difference between the two sets of equations increases significantly. Similarly to what was previously found, the actual data points do not follow the Expected Value at all until they merge around (N,V). However, something interesting in this case is that the data seems to fall between two standard deviations of the expected value earlier than the PubMed abstracts. However, it still only merges very close

to the end. This shows that, just like with the PubMed abstracts, the expected value equation does not accurately represent the actual data when not using the entire data set as a sample. The change in when the actual data merges with the expected value equation also seems to show something regarding how language learning models create text compared to humans.

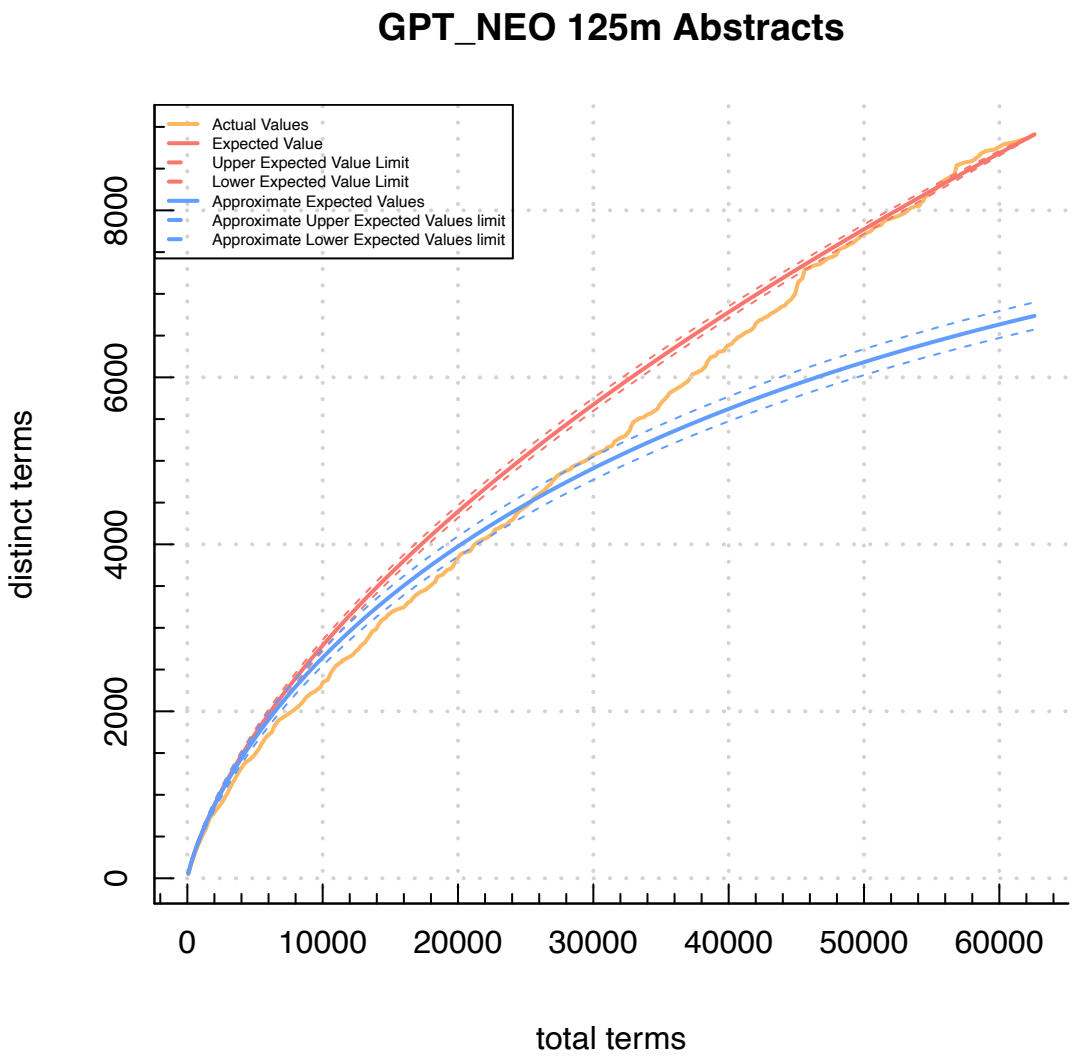

Figure 2: Graph of the expected value equations and the actual data for 250 Abstracts created by GPT-Neo with 125 million parameters.

Thirdly, I worked with abstracts created by GPT-Neo with 1.3 billion parameters. This graph seems to be a middle ground between the previous GPT-Neo graph and the PubMed graph. Just like the previous two graphs, the binomial approximation of the expected values and variance start out similar to one another but begin to diverge as the number of words increases. Also, like the previous two graphs, the line of the actual data does not seem to

merge with the expected values until the last few documents. However, compared to the GPT-Neo abstracts with 125 million parameters, the actual data merges with the expected value later with the abstracts with 1.3 billion parameters, but it also merges earlier than the PubMed abstracts.

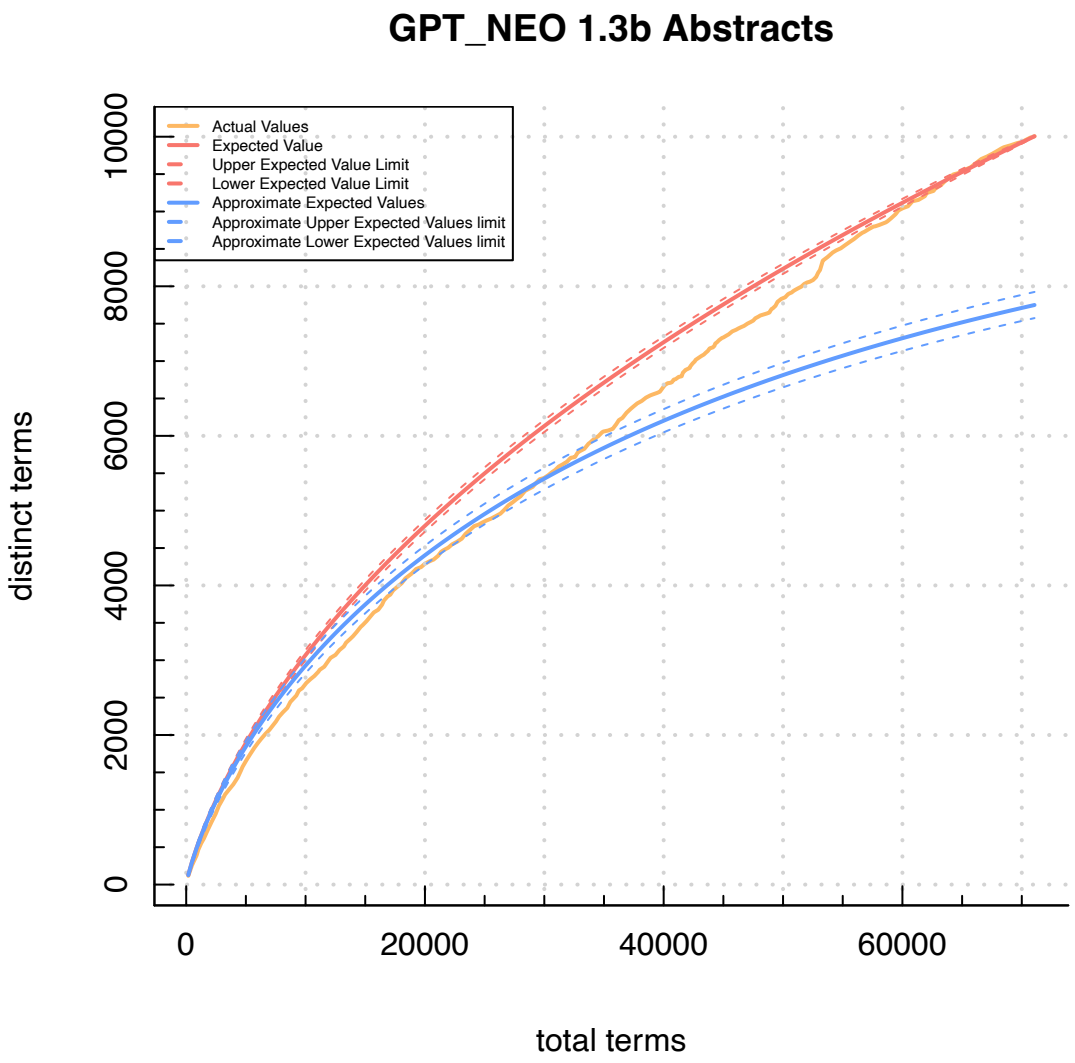

Figure 3: Graph of the expected value equations and the actual data for 250 Abstracts created by GPT-Neo with 1.3 billion parameters.

Finally, I did the exact same thing with abstracts created by GPT-Neo with 2.7 billion parameters. This one provided the most intriguing results, as it ended up being the closest to the PubMed abstracts out of the three GPT-Neo data sets. Like all the previous graphs, the two expected values and variance equations diverge as the number of words increases. Similar to the previous models, the actual data only approaches the expected value equation as the total terms approach V. Compared to the other three GPT-Neo graphs, this

graph has the two lines merging at the latest points. However, it is still earlier than the PubMed data.

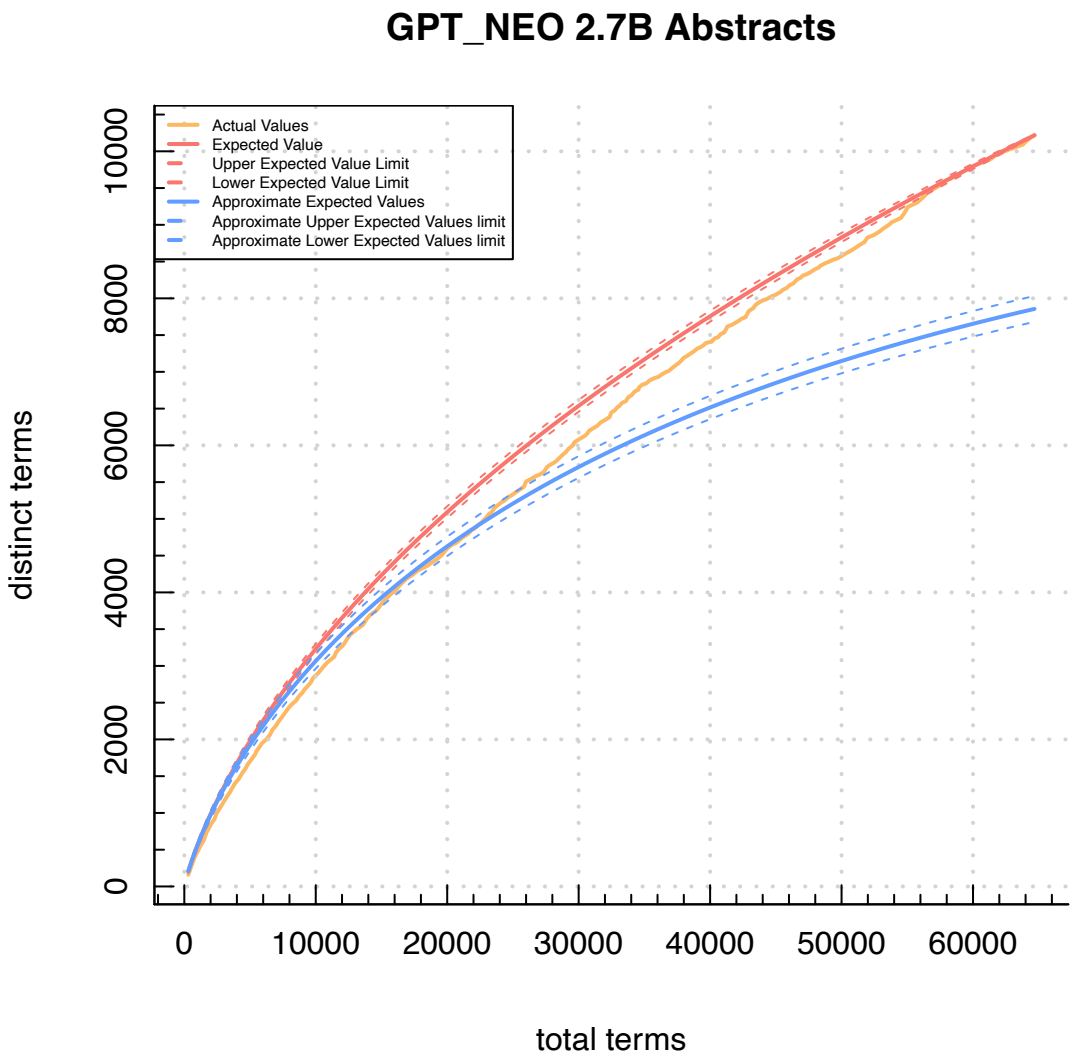

Figure 3: Graph of the expected value equations and the actual data for 250 Abstracts created by GPT-Neo with 2.7 billion parameters.

Conclusions:

So, before reaching the main conclusion, it is important to see how changing the number of parameters in the GPT-Neo data affects the estimation of Heaps' Law. As mentioned through the analysis, the three different GPT-Neo models all had their actual data fall within two standard deviations of the expected values earlier than the PubMed data. However, as the number of parameters increased, this point of intersection occurred later. This makes sense, as researchers have found that the more parameters put into a language

learning model such as GPT-Neo, the more human-like the text becomes. In theory, if there was a language model that took in infinite parameters and graphed Heaps' law, I believe we would get a graph nearly identical to the PubMed abstracts.

Another exciting component of the models is that for all of them, the binomial approximation of the expected value began to differ extremely as the number of total terms increased. This shows that the assumptions made when creating the formula do not hold true, which makes sense as there is only a finite number of words. Also, a key difference between a binomial distribution and a hypergeometric distribution is that a hypergeometric distribution does not have replacement, while a binomial distribution does. Therefore, I can conclude that as the total number of terms approaches N, the effect replacement has on the expected value and variance equations increases, which would explain the difference between the two sets of equations.

Overall, it seems that my hypothesis is correct. The expected value equation does not accurately predict the size of the vocabulary. For all of the four sets, the actual data does not actually fall within 2 standard deviations above or below the data until the last few documents, with that number going down the more human-like the abstracts are. This means that we cannot use the expected value equation to determine the minimum number of documents needed to determine the parameters of Heaps' law accurately. If we did, then we would get the total number of documents for the PubMed data and practically all of the documents for the three GPT-Neo sets.

Possible Improvements:

The main possible improvement would be that instead of going through each document in order, Heaps' law's expected value equation randomly selects the documents used. However, this should not affect our conclusion, as even if a later document gets selected earlier, there are still the same number of documents each time. For any three abstracts, the number of terms and vocabulary size should remain similar, and the change in documents should also affect the expected value and variance, which, similarly, means the conclusion should remain the same.

Sources:

Chacoma A. and Zanette. "Heaps' Law and Heaps' functions in tagged texts: evidence of their linguistic relevance" https://royalsocietypublishing.org/doi/full/10.1098/rsos.200008

Lai, Uyen, Gurjit Randhawa and Paul Sheridan. "Heaps' Law in GPT-Neo Large Language Model Emulated Corpora.