



For this assignment, we will use only some of the essays of Hamilton (1, 6, 7, 8, 9, 11, 12, 13, 15, 16, 17, 21, 22, 23, 24, 25, 26, 27, 28, 29), and all of the essays of Madison (10, 14, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 58). we will hold out three essays (9, 11, 12) of Hamilton and three essays (47, 48, 58) of Madison for testing in Task 3. The rest of the essays (17 Hamilton essays and 12 Madison essays) will be used for training in Task 1.

This special dataset creates all the data to be created and the data to be used for the test.

### 3 Task 1: Building Language Models

In this section, the words used by the authors to create n-gram models; includes parsing, tokenizing, remove punctuations. The tokenizing process for creating the models varies according to the model type.

Firstly, essay objects are created by parsing according to the ngram model and then the essay objects are sent to the corresponding author model to create n-gram models.

#### 3.1 Unigram Model

The only process that needs to be done when creating a unigram model is to separate punctuation marks from words. For this reason, only BagOfWords structure has been created that has uniq words and contains the frequencies of words. It will also be described in Laplace Smoothing Task 3, which will be used for probability calculations during testing. For example for all models punctuations:

"I want, to go: home." —after tokenizing—> "I want , to go : home ."

#### 3.2 Bigram Model

During the creation of the Bigram model, the Tokenizing process is required. Because it is important to have words at the beginning or end of the sentence. Therefore, for markings that terminate the sentence (".", "!", "?"), The start and end tokens are replaced by adding. For example :

"I want to go home" —after tokenizing—> "<s> I want to go home </s>"

"Are you sure that?" —after tokenizing—> "<s> Are you sure that </s> ?"

#### 3.3 Trigram Model

During the creation of the Trigram model, the Tokenizing process must also be performed. Because it is important to have words at the beginning or end of the sentence. Therefore, for the markings that terminate the sentence (".", "!", "?"), **Two start** and **two end** tokens are added and replaced. For example:

"I want to go home" —after tokenizing—> "<s> <s> I want to go home </s> </s>"

"Are you sure that?" —after tokenizing—> "<s> <s> Are you sure that </s> </s> ?"

Table 1: Uniq Word Counts for Hamilton

Unigram	Bigram	Trigram	After Tokenizing and Replacement ->	Unigram	Bigram	Trigram
5963	21069	30696		4414	19556	32123

Table 2: Uniq Word Counts for Madison

Unigram	Bigram	Trigram	After Tokenizing and Replacement ->	Unigram	Bigram	Trigram
5579	20204	29867		4051	18445	30919

As can be seen the given tables, the models are tokenizing and replacement operations changes the uniq word counts. This is directly effective on the results we will obtain from the models. In addition, as seen in Model Vocabulary, the number of uniq words increases as the number of N grows, which can lead to more accurate results. but if there is not enough train data, increasing the N number may cause negative results and may result in incorrect results. Therefore, we will try to determine the most appropriate model for our datas by using the **Perplexity** calculation in Task 3.

## 4 Task 2: Automatically Generating Essays

In this part of the assignment, we will try to generate essay for each author using the models we have obtained. To do this, we will use uniq words we keep in BagOfWords. In order for these randomly generated essays to be consistent, we will look at the relationship between the words. In doing this, we will create word relationship probability tables for bigram and trigram models and we will randomly add one to our essay. It is also designed to stop adding essay e words as soon as the number of words in the essay reaches 30 or any token that ends the sentence.

### 4.1 For Generate Unigram

Since there is no contextual relation between the words in Unigram structure, we will create a table with a total probability of 1 by collecting the probabilistic of all words. Then we will create a random number between 0 and 1 and we will select the corresponding word in the table and add it to our essay. When creating this structure, we will be able to identify the initial and end probability values of each word and determine the corresponding value at different intervals. For example :

bagOfwords : { "I": 3 , "want": 4 , "go" : 2 }  
cumulativeDict : { "I": { "start": 0 , "end":  $\frac{3}{9}$  } , "want": { "start":  $\frac{3}{9}$  , "end":  $\frac{5}{9}$  } , "go": { "start":  $\frac{5}{9}$  , "end":  $\frac{9}{9}$  } }

Example Generated Essays:

\*\*\*\*\*

This Essay generated for: HAMILTON and use with UNIGRAM model →

**of fatal this penetration blush that resolutions to politic more be of suspicious the of advice apply in supreme of rebellions every , exemption prerogative most proper an of**

Test Result:

Hamilton's generated test for UNIGRAM WIN:[HAMILTON] Hamilton Prob: 270.21569 Madison Prob: 278.74627

\*\*\*\*\*

This Essay generated for: MADISON and use with UNIGRAM model →

**in occasion convention confidence , but unjust , have given it each repaired awkward is ?**

Test Result:

Madison's generated test for UNIGRAM WIN:[HAMILTON] Hamilton Prob: 427.10133 Madison Prob: 427.69237

\*\*\*\*\*

### 4.2 For Generate Bigram

There is a contextual relationship between the words in the Bigram structure, so we will create a table with a total probability of 1 by collecting the probabilities of words that may come after the last word. Then we will create a random number between 0 and 1, and we will add the corresponding word in the table and add it to our composition. When creating this structure, we will be able to define the first and last probability values of each word and determine the corresponding value at different intervals. For example :

Actual Example essay : " Thank a lot but I"  
bagOfwords with start "I" : { "I want": 2 , "I go": 3 , "I to": 2 }  
cumulativeDict : { "want": { "start": 0 , "end":  $\frac{2}{7}$  } , "go": { "start":  $\frac{2}{7}$  , "end":  $\frac{5}{7}$  } , "to": { "start":  $\frac{5}{7}$  , "end":  $\frac{7}{7}$  } }

Example Generated Essays:

\*\*\*\*\*

This Essay generated for: HAMILTON and use with BIGRAM model →

**<s> this power would be proportioned to the government to shorten an army and the nations than they gratify their declamations and tendency to his family compact between the expedient of**

Test Result:

Hamilton's generated test for BIGRAM WIN:[MADISON] Hamilton Prob: 443.23033 Madison Prob: 440.66789

\*\*\*\*\*

This Essay generated for: MADISON and use with BIGRAM model →

**<s> but with deliberation and better administration , ' it was laid under the instrument </s> .**

Test Result:

Madison's generated test for BIGRAM WIN:[MADISON] Hamilton Prob: 686.21348 Madison Prob: 682.28677

\*\*\*\*\*

### 4.3 For Generate Trigram

There is a contextual relationship between the words in the Trigram structure, so we will create a table with a total probability of 1 by collecting the probabilities of words that may come after the last **two word**. Then we will create a random number between 0 and 1, and we will add the corresponding word in the table and add it to our composition. When creating this structure, we will be able to define the first and last probability values of each word and determine the corresponding value at different intervals. For example :

Actual Example essay : " Thank a lot but I"

bagOfwords with start "but I" : { "but I want": 2 , "but I go": 3 , "but I to": 2 }

cumulativeDict : { "want": { "start": 0 , "end":  $\frac{2}{7}$  } , "go": { "start":  $\frac{2}{7}$  , "end":  $\frac{5}{7}$  } , "to": { "start":  $\frac{5}{7}$  , "end":  $\frac{7}{7}$  } }

Example Generated Essays:

\*\*\*\*\*

This Essay generated for: HAMILTON and use with TRIGRAM model ->

**<s> <s> [ 1 ] if the federal government , you are called upon to deliberate on a proportion , less likely to take an enlarged scale </s> </s> .**

Test Result:

Hamilton's generated test for TRIGRAM WIN:[MADISON] Hamilton Prob: 448.89230 Madison Prob: 447.13198

\*\*\*\*\*

This Essay generated for: MADISON and use with TRIGRAM model ->

**<s> <s> it is an object of the state will no doubt provide in the establishment of courts for the system has been already sufficiently vindicated and explained </s> </s> .**

Test Result:

Madison's generated test for TRIGRAM WIN:[MADISON] Hamilton Prob: 897.78460 Madison Prob: 894.26397

\*\*\*\*\*

### 4.4 Model Differences For Generating Essays

After creating a random essay, we see that as the number of n-grams grow, the essay produced becomes more logical sentences. This is due to the fact that the relationship between words more forceful random words to ensure that the words to be meaningful. When we look at the essays derived from Unigram, we see that random words with no meaning are selected. It is possible to see that the punctuation marks in the sentence are more accurate as the number of n -gram increases.

In addition, when we test the produced essays, we get the result written by the other author even if they are produced from their own word groups. One of the reasons for this is that the train data is not rich and the other is that the words and language structure of the authors are very similar. It is also possible to observe that the probability values are quite close.

## 5 Task 3: Classification and Evaluation

In this section of the assignment, we will try to determine who can belong to whom the articles which are not known to belong to whom by using the models we created for the authors. In doing so, we will perform the probability calculation using the word frequencies in the models we created.

We need to pay attention to the fact that Train data is not **unseen** in the probability of zero words to ensure that the probability to make the calculation of the accuracy. We will use **Laplace smoothing** to overcome this problem.

Laplace smoothing provides very low probability values of unexceptional words. Briefly takes some of the other possibilities, adds to zero probabilities. This helps to produce accurate results. The general formula for calculating word possibilities is as follows:

Original for Bigram:

$$P(w, w_{n-1}) = \frac{C(w_n | w_{n-1})}{c(w_n)}$$

New for Bigram :

$$P(w. | w_n) = \frac{c(w. | w_n) + 1}{c(w_{n-1}) + V}$$

After calculating the probabilities in this way, we collect logarithmically. If we do not collect logarithm, we get very small values. Then, we calculate the value of **Perplexity** with the formula given below:

$$PP(W) = 2^{-\frac{1}{N} \sum_{i=1}^N \log_2 P(w_1 w_2 \dots w_N)}$$

In general, perplexity is a measurement of how well a probability model predicts a sample. In the context of Natural Language Processing, perplexity is one way to evaluate language models. But we must not forget that perplexity gives an **inverse probability** calculation.

## 5.1 Classification Results and Perplexity Analysis

Results For Hamilton Test Essays :

9.txt UNIGRAM WIN:[HAMILTON] Hamilton Perplexity: 449.81578 Madison Perplexity: 453.82573  
 11.txt UNIGRAM WIN:[HAMILTON] Hamilton Perplexity: 474.49998 Madison Perplexity: 514.49440  
 12.txt UNIGRAM WIN:[HAMILTON] Hamilton Perplexity: 466.78682 Madison Perplexity: 492.35673  
  
 9.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3658.19594 Madison Perplexity: 3517.84272  
 11.txt BIGRAM WIN:[HAMILTON] Hamilton Perplexity: 3869.33691 Madison Perplexity: 4051.27540  
 12.txt BIGRAM WIN:[HAMILTON] Hamilton Perplexity: 3739.79222 Madison Perplexity: 3900.75433  
  
 9.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12768.27732 Madison Perplexity: 12324.94210  
 11.txt TRIGRAM WIN:[HAMILTON] Hamilton Perplexity: 13120.97157 Madison Perplexity: 13508.39863  
 12.txt TRIGRAM WIN:[HAMILTON] Hamilton Perplexity: 12909.57700 Madison Perplexity: 13128.72663

Results For Madison Test Essays :

47.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 432.95533 Madison Perplexity: 375.46700  
 48.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 431.52763 Madison Perplexity: 384.75164  
 58.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 392.86570 Madison Perplexity: 362.71797  
  
 47.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3540.05435 Madison Perplexity: 2916.00913  
 48.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3639.11473 Madison Perplexity: 3129.64151  
 58.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3467.48202 Madison Perplexity: 2962.63573  
  
 47.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13631.28248 Madison Perplexity: 11866.63114  
 48.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13572.44291 Madison Perplexity: 12355.72567  
 58.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13536.16909 Madison Perplexity: 12141.76165

Results For Unknown Author Test Essays :

49.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 385.16207 Madison Perplexity: 360.30581  
 50.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 455.37239 Madison Perplexity: 425.36627  
 51.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 384.38708 Madison Perplexity: 348.69196  
 52.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 417.18904 Madison Perplexity: 379.43894  
 53.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 429.04186 Madison Perplexity: 397.95632  
 54.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 376.40093 Madison Perplexity: 342.11296  
 55.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 438.34637 Madison Perplexity: 410.14392  
 56.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 392.75314 Madison Perplexity: 361.22101  
 57.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 444.32690 Madison Perplexity: 414.76893  
 62.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 448.66343 Madison Perplexity: 406.66878  
 63.txt UNIGRAM WIN:[MADISON] Hamilton Perplexity: 439.14716 Madison Perplexity: 401.98879  
  
 49.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3112.16007 Madison Perplexity: 2737.20678  
 50.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3538.74622 Madison Perplexity: 3154.91640  
 51.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3277.67072 Madison Perplexity: 2808.01273  
 52.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3272.74425 Madison Perplexity: 2756.77663  
 53.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3647.96963 Madison Perplexity: 3137.71292

54.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3376.83217 Madison Perplexity: 2827.30816  
 55.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3938.63607 Madison Perplexity: 3370.93183  
 56.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3345.80124 Madison Perplexity: 2880.40336  
 57.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3728.00633 Madison Perplexity: 3297.91031  
 62.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3662.70964 Madison Perplexity: 3124.32121  
 63.txt BIGRAM WIN:[MADISON] Hamilton Perplexity: 3763.54957 Madison Perplexity: 3264.38667

49.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12071.42856 Madison Perplexity: 11124.05487  
 50.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12777.30494 Madison Perplexity: 11794.21237  
 51.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13038.00844 Madison Perplexity: 11748.14961  
 52.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12575.68370 Madison Perplexity: 11092.26659  
 53.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13700.49926 Madison Perplexity: 12459.69933  
 54.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13329.99381 Madison Perplexity: 11758.31078  
 55.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 14429.07767 Madison Perplexity: 12879.87512  
 56.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 13328.06757 Madison Perplexity: 12074.56933  
 57.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12935.03269 Madison Perplexity: 11983.26105  
 62.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 12931.33001 Madison Perplexity: 11670.33018  
 63.txt TRIGRAM WIN:[MADISON] Hamilton Perplexity: 14053.71052 Madison Perplexity: 12842.78394

In normal conditions, we expect bigram and trigram results to be more accurate. But due to the fact that train data is not rich, we can see that we get the best result in Unigram. When we look at perplexity values, we can see this clearly. The lowest perplexity shows the best model to us. Here we can see that Unigram has the lowest perplexity. Unigram is therefore the right choice for our data.


The reason why all of them are classified as Madison for unknown essay is that it is probably due to the use of punctuation marks or stop-words in the test data.

In this assignment, it is normal for us to just look at the semantic relations and make this sort of result because the train data is not rich. If we were doing Lexical analysis, we could have more accurate results.

As a result, we have a sufficient model for Startup, but using different processes will allow us to get more accurate results, especially when we have less data.

## HYPNOSIS AND NLP PODCAST

*The Failure into Feedback Loop - NLP Technique Explanation & Session*



LEARN HOW TO TAP INTO THE POWER OF YOUR MIND

Donald Currie, Registered Psychotherapist