Author attribution using neural networks

Anthony Pasqualoni
June 27, 2006

Introduction

This project is an experiment in stylometry. Stylometry is the study of linguistic style; it uses quantitative measurements, often lexical in nature, to capture the style of a particular author [1,2].

Previous research has shown that neural networks are effective in stylometry. Multilayer perceptrons using back propagation have been used to identify authors of works by Shakespeare, Fletcher, and Marlowe, and to identify the authors of *The Federalist Papers* [2]. In the former case, five ratios of word pairs were used as discriminants; the latter used letter pair frequencies.

For this project a neural network was designed to determine the authors of English sonnets written by William Shakespeare and other poets of the same period. The network is trained to attribute a given sonnet to either Shakespeare or a poet from a set of three authors. During the final test, on average over 90 out of 100 sonnets were correctly attributed.

The project comprises two components: a Python script for text processing and feature measurement, and a neural network written in C.

Text processing and feature measurement

The Python script processes two ASCII text files obtained from Project Gutenberg, one comprising a collection of sonnets written by William Shakespeare (1564-1616), the other a selection of sonnets from three other poets: Edmund Spenser (1552?-1599), William Smith (fl. 1596), and Bartholomew Griffin (-1602). Text processing includes the following: removing leading white space, converting all letters to lower case, and removing punctuation and non-alphanumeric characters.

To produce input data for the neural network, the Python script measures three lexical features, or discriminants, from the texts: word pair counts both horizontally (same line) and vertically (between adjacent lines, divided into columns one word wide), and individual word counts. Originally, letter pair counts were also used, but excluding this measurement increased the accuracy of the neural network.

Feature measurement is restricted to the training sets. The training sets comprise fifty sonnets from Shakespeare, and a total of fifty sonnets from Spenser, Smith, and Griffin. This is in line with the assumption that the authors of the sonnets used in the final test are unknown. The Python script compiles lists of high and low frequency words and word pairs from the training sets. Frequency is a measurement of occurrence relative to Shakespeare's sonnets, with high frequency words occurring more often in Shakespeare.

The procedure for compiling each list is as follows. Note that the term 'discriminants' refers to the three lexical features described above: individual words and vertical and horizontal word pairs.

> **1.** Count all discriminants in the training set of sonnets written by Shakespeare and the training sonnets written by Spenser, Smith, and Griffin.
>
> **2.** Determine the difference in frequency between the two training sets for each discriminant by dividing the amount of occurrences in Shakespeare by the amount of occurrences in the other author set.
>
> **3.** If the difference is above a pre-defined threshold value, add the discriminant to the high frequency list. Otherwise, if it is below a specified threshold value, add the discriminant to the low frequency list.

Threshold values for determining inclusion in the discriminant lists were chosen by running the Python script over 2000 times with random threshold values applied to the training sets. The neural network was used to determine the error rate on both the training data and the test data for each set of random threshold values. The values that produced the highest average accuracy in the network using the test set were then chosen as the final threshold values.

As an example of feature measurement, shown below are some of the individual words that occur frequently in the training set comprising Shakespeare's sonnets, but rarely or not at all in the training sonnets by Spenser, Smith, and Griffin. Since the threshold value for high frequency individual words is set to 3.47, these words are included in the high frequency word list.

	Occurrences in Shakespeare:	Occurrences in Spenser et al.:	Difference:
self	23	0	23.00000
own	14	1	14.00000
time	23	2	11.50000
times	11	0	11.00000
even	9	0	9.00000
eye	18	2	9.00000
an	8	1	8.00000
old	8	1	8.00000
say	8	1	8.00000
against	7	1	7.00000

The last column is calculated simply by dividing the amount of occurrences in Shakespeare by the occurrences in Spenser, Smith, and Griffin. In the case where the latter is zero, the difference is set to the amount of occurrences in Shakespeare. For example, in the training sets, the word 'own' occurs 14 times more frequently in Shakespeare than it does in Spenser, Smith, and Griffin. Since these words occur frequently in Shakespeare's sonnets but rarely in the others, they are included in the high frequency word list.

Shown below are words included in the low frequency word list. These words occur often in sonnets by Spenser, Smith, and Griffin, but rarely or not at all in Shakespeare's sonnets:

	Occurrences in Shakespeare:	Occurrences in Spenser et al.:	Difference:
unto	1	21	0.04762
cruel	1	11	0.09091
pity	1	11	0.09091
fire	1	8	0.12500
ever	2	15	0.13333
never	2	13	0.15385
cause	1	6	0.16667

```
  pride  1                    6                      0.16667
  sighs  1                    6                      0.16667
     ye  1                    6                      0.16667
```

Neural Network

The neural network is a multilayer perceptron with two hidden layers of twenty nodes each. In addition, there are six input nodes: two for high and low frequency word counts, two for high and low frequency horizontal word pair counts, and two for high and low frequency vertical word pair counts. There is one output node, with values close to +1 indicating the input vector represents a sonnet written by Shakespeare, and values close to -1 indicating a sonnet written by another author.

Synaptic weights and biases are initialized to random values between -1 and +1 and the network is trained using back propagation. Node output is determined using inputs from the previous layer modified by weights and a bias. A sigmoid function is used for the activation function. The learning rate was chosen using trial and error.

Code for the design of the network and the back propagation functions is based on the algorithm described in [3].

After training data comprising discriminant counts for both author sets are loaded from a file produced by the Python script, the network begins the training phase with a maximum of 10,000 iterations. With each iteration, node outputs are calculated and weights are adjusted using back propagation. Approximately every 1,000th iteration triggers a test on all training patters. If there are no failures, the training loop ends. Otherwise, training continues until the maximum amount of iterations is reached or until all training patterns are classified correctly.

Once training is complete, the network is tested with fifty sonnets from Shakespeare, and fifty from the other author set. None of the sonnets used in the final test are included in the training sets.

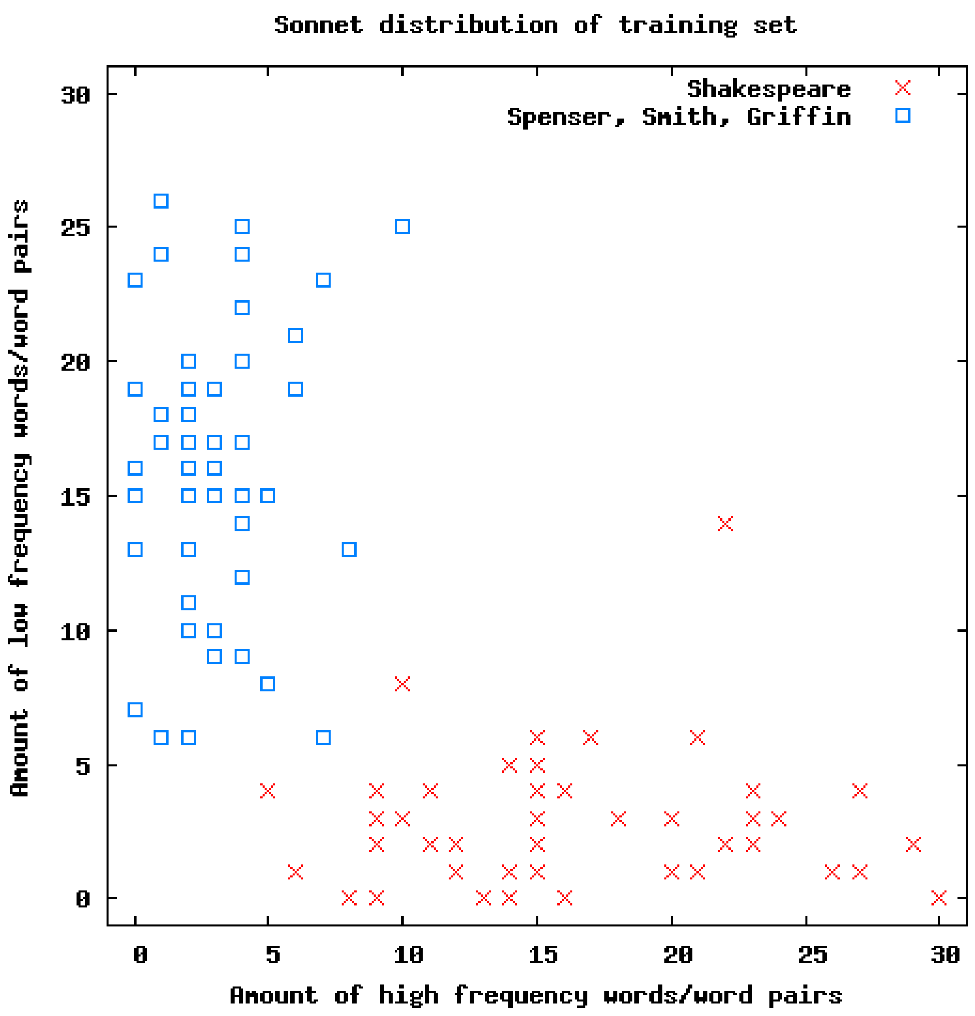
Results

Shown below is the output of the neural network. The network code was compiled using the -O3 option in GCC.

```
Hidden layers:      2
Nodes per layer:    20
Maximum iterations: 10000
Learning rate:      0.050000

Input data from 200 sonnets:
   Training: 50 sonnets by Shakespeare; 50 by Spenser, Smith, & Griffin
   Test:     50 sonnets by Shakespeare; 50 by Spenser, Smith, & Griffin

Run:     Iterations:    Training failures:    Test failures:
  1            1001              0 of 100          7 of 100
  2            5001              0 of 100          7 of 100
  3            5001              0 of 100          7 of 100
  4            5001              0 of 100          6 of 100
  5            5001              0 of 100          8 of 100
  6            5001              0 of 100          7 of 100
  7            1001              0 of 100          6 of 100
  8            6001              0 of 100          7 of 100
  9            5001              0 of 100          8 of 100
 10            2001              0 of 100          6 of 100

Average accuracy in identifying author after training: 93.10%
```

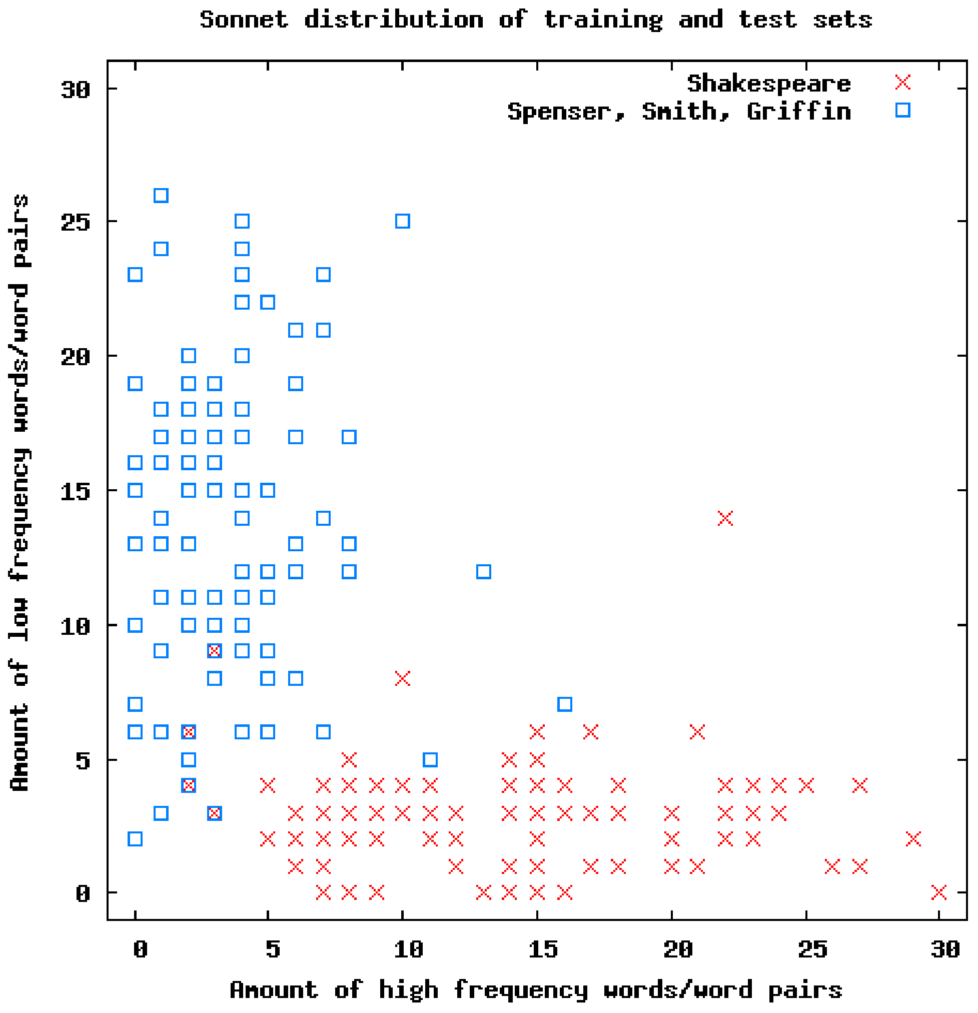
High accuracy is achieved because the input data form definable clusters with little overlap, as shown in the graphs below.

The following graph shows word and word pair counts for each sonnet in the training sets. Each point in the graph represents a sonnet, or more than one sonnet in the few cases where coordinates are equal. The horizontal axis represents the amount of discriminants in the high frequency list, and the vertical axis measures the amount of discriminants in the low frequency list.



As expected, most of the sonnets by Shakespeare (represented by red points) include more high frequency discriminants than low. Likewise, the sonnets by Spenser, Smith, and Griffin exhibit more low frequency discriminants. With the right threshold values, the sonnets arrange themselves into definable clusters. While not linearly separable, the clusters can be divided with a curve. This explains why the network can be trained to recognize training patterns without error.

Equally important is the question of whether the training data accurately reflect the entire data set. If the sonnets from the test sets do not reflect the training sonnets, the network will fail. The next graph, shown below, includes sonnets from both the training sets and the test set.



This graph shows that there is overlap between the two author groups; boundaries are less defined and some points represent sonnets from both author groups. As a result, it is not possible to define a curve that separates the two clusters. This explains why the network is unable to classify the test patterns without error (typically there are about 5-10 errors out of 100 sonnets). However, aside from the border area where overlap occurs, two definable clusters have formed, both similar to the training set clusters. This shows that the training data accurately reflect the entire data set with few exceptions. The result is that the neural network is able to classify sonnets with an average accuracy above 90%.

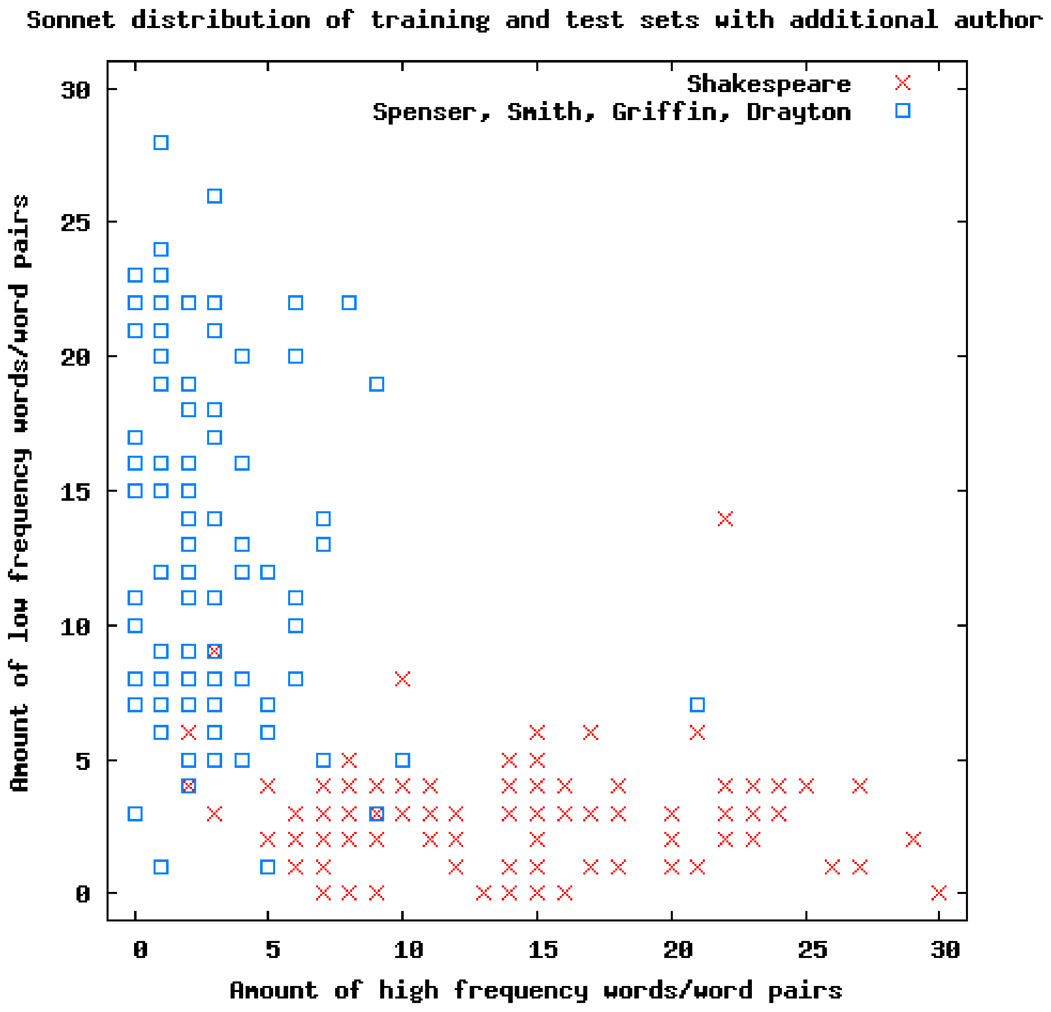
Limitations

As explained above, threshold values for compiling word and word pair lists were chosen using results from the neural network, with thresholds resulting in the highest accuracy being selected as the final values. In a real world application, where the author of a text is truly unknown, it would not be possible to perform this process since the accuracy of the test results would not be able to be determined. However, in some cases the author of a text is suspected, in which case a probabilistic measurement of accuracy may be possible. In any case, this experiment has shown that it is possible to obtain high accuracy rates in authorship attribution using simple lexical measurements, provided enough data are provided and a limited author set is used.

Another limitation is that the code is fine-tuned for recognizing patterns in a particular set of authors: William Shakespeare as compared with Edmund Spenser, William Smith, and Bartholomew Griffin. The neural network's accuracy is reduced when other authors are added to the data set. For the sake of comparison, sonnets by another author, Michael Drayton (1563-1631), were added to the collection of sonnets by Spenser, Smith, and Griffin, and high and low frequency discriminant lists were compiled without changing the threshold values. Shown below is the network's output:

```
Hidden layers:      2
Nodes per layer:    20
Maximum iterations: 10000
Learning rate:      0.050000

Input data from 200 sonnets:
   Training: 50 sonnets by Shakespeare; 50 by Spenser, Smith, Griffin, & Drayton
   Test:     50 sonnets by Shakespeare; 50 by Spenser, Smith, Griffin, & Drayton

Run:     Iterations:    Training failures:    Test failures:
  1          10000                2 of 100        18 of 100
  2          10000                2 of 100        18 of 100
  3          10000                2 of 100        20 of 100
  4          10000                2 of 100        21 of 100
  5          10000                2 of 100        19 of 100
  6          10000                2 of 100        22 of 100
  7          10000                1 of 100        20 of 100
  8          10000                4 of 100        19 of 100
  9          10000               50 of 100        50 of 100
 10          10000                2 of 100        20 of 100

Average accuracy in identifying author after training: 77.30%
```

The following graph shows why accuracy is reduced with the additional author. As above, it illustrates the distribution of sonnets in terms of high and low frequency discriminants for both training and test sets:



With sonnets from the additional author, clusters are less defined. This is reflected in the network's output: there are failures in the training phase and average accuracy achieved in the test phase is only 77.30%.

Since the threshold values that produced the above results were fine-tuned for three authors, accuracy can be improved in this case by adjusting the threshold values to suit the new data set. Using a random search, new threshold values were discovered that produced improved results, and the average accuracy of the neural network increased to 84.40%. This shows that even with a larger number of authors, enough clustering occurs to allow pattern recognition with some level of accuracy. Also, it may be possible to attain higher accuracy by measuring additional features (e.g., punctuation or function words) in addition to changing threshold values for the discriminant lists.

Finally, another limitation is that the spelling used in at least some of the sonnets differs from the original. The following two lines are from Sonnet 100 from the 1609 edition of *Sonnets* by William Shakespeare [4]:

> Giue my loue fame faster then time wasts life,
> So thou preuentst his sieth, and crooked knife.

The same lines from the Project Gutenberg text [5] which was used in the data set:

> Give my love fame faster than Time wastes life,
> So thou prevent'st his scythe and crooked knife.

Although only words and word pairs are used as discriminants, using the original spelling may have some effect in the identification of words. Also, the difference in spelling may be one reason why letter pair counts were not effective in pattern recognition.

Future research

Possible avenues of exploration include the following:

- Use a genetic algorithm to evolve a feature set (in addition to words and word pairs) or parameters for feature extraction.
- Experiment with neural networks other than multilayer perceptrons.
- In addition to sonnets, apply the neural network to texts in other formats, such as essays, articles, or fiction.
- Compare the neural network with other methods of pattern recognition.

Data files and code

Data files:

shakespeare.txt
spenser-smith-griffin.txt
spenser-smith-griffin-drayton.txt

Python scripts and neural network code:

run.py
extract.py
nn-nlp.c

References

[1] Stylometry. (2006, June 17). In Wikipedia, The Free Encyclopedia. Retrieved 13:16, June 27, 2006, from http://en.wikipedia.org/wiki/Stylometry.

[2] McEnery, T., & Oakes, M. (2000). Authorship Identification and Computational Stylometry. In Dale, R., Moisl, H., & Somers, H. (Ed.), *Handbook of Natural Language Processing* (pp. 545 - 562). New York: Marcel Dekker.

[3] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. New Jersey: Prentice-Hall.

[4] Shakespeare, W. *Sonnets* (1609 Edition). Electronic Text Center, University of Virginia Library. Retrieved June 27, 2006 from http://etext.lib.virginia.edu/toc/modeng/public/ShaSonQ.html

[5] Shakespeare, W. *Shakespeare's Sonnets*. Project Gutenberg. Retrieved June 27, 2006 from http://www.gutenberg.org/etext/1041