Hidden Markov Models for Collaborative Works

Shane McQuarrie

1 Introduction

Suppose that you have just written a best-selling novel with your colleague Jack. Critics say that the novel starts a little slowly, but that it's worth it to trudge through the more boring and stale parts of the book because of the exciting climax and cliff-hanger ending. As the novel gains popularity and publicity, more and more readers and critics are asking the obvious question: which one of the authors wrote the climax of the book? You and Jack both know that Jack and his somewhat mediocre writing style are responsible for the more unimaginative parts of the book, and it was your ingenuity and creativity throughout the climax that made the book successful. However, Jack is now claiming that he was the sole writer for the climax, and that you are responsible for the book's weaker sections. Can you prove him wrong?

This scenario describes a common problem[1] in *Stylometry*, the statistical study of linguistic style: is it possible to determine the author of a written work? Most text classification and authentication algorithms examine documents of unknown authorship with the goal of determining the singular author. This is a well-studied problem, with many effective solutions via principal component analysis (PCA) and neural networks. What if, however, a document has two or more authors (such as you and Jack)? We examine the case in which the several authors of a collaborative work are already known, but it is not known which author wrote what sentences. The question is this: given sufficient training data, is it possible to partition a collaborative document by its authors? We approach the problem with hidden Markov models (HMM) to compare the style of the collaborative document with the style of existing works by each author.

2 Data and Preliminary Analysis

Data Sets

Data for this project is readily available. `http://www.gutenberg.org` has many free large collections of texts from famous authors. In addition, Python's `nltk` module (natural language toolkit) has many documents available specifically for natural language processing.

We make use of the following data sets:

[1]Other current computational linguistics problems include plagiarism detection, contextual authentication, and authorship attribution in instant messaging.

Description	Source
The complete poetical works of Edgar Allen Poe	Gutenberg
Ronald Reagan's State of the Union addresses	`nltk.corpus.state_union`
Franklin D. Roosevelt's presidential inaugural addresses	`nltk.corpus.inaugural`
Eight of William Shakespeare's plays	`nltk.corpus.shakespeare`
The complete collection of William Shakespeare's sonnets	Gutenberg
Yoda's lines from Star Wars episodes I, II, III, V, and VI	Screenplays

We will consider the two Shakespeare data sets separately, since sonnets and the plays have very different writing structures (for instance, plays have stage directions).

Word Count

We begin with a basic analysis of the source material. *Stop words* are common words that don't reveal much about an author's vocabulary or style. For example, the words 'am', 'be', 'her', 'only', 'now', 'that', 'with', 'you', and just under 200 other common words are contained in the `nltk` module's list of stop words. Excluding these words, the 20 most commonly used words by each author (in order, ignoring punctuation and case) are:

Author	20 Most Frequent Words
Poe	upon, thy, one, thou, thee, love, would, yet, poem, heart, like, bells, heaven, o, us, soul, thus, night, may, within.
Reagan	us, must, people, america, government, years, year, american, federal, tax, freedom, new, world, congress, one, economic, time, work, peace, future.
Roosevelt	nation, people, government, democracy, shall, us, men, must, life, know, spirit, upon, new, may, years, national, world, every, states, good.
Shakespeare (plays)	thou, shall, o, thy, lord, thee, come, good, enter, love, antony, well, let, caesar, would, hamlet, go, hath, man, upon.
Shakespeare (sonnets)	thy, thou, thee, love, doth, mine, shall, eyes, sweet, time, beauty, yet, art, o, heart, thine, hath, fair, make, one.
Yoda	must, dark, force, jedi, side, fear, much, go, training, see, mind, ready, path, strong, trained, help, one, master, apprentice, future.

This at least gives us a feel for which authors are more similar to others, based on our data sets. Unsurprisingly, the poets Poe and Shakespeare appear to be similar, and the politicians Roosevelt and Reagan are likewise comparable. Yoda doesn't appear to be very similar to any of the other authors, due to his unique vocabulary.

Excluding all words of length 4 or less yields more interesting results.

Author	20 Most Frequent Words (except words shorter than 5 characters)
Poe	heaven, within, beauty, spirit, shadow, little, things, effect, poetry, nothing, politian, flowers, length, thought, though, indeed, nevermore, dreams, around, without.
Reagan	people, america, government, american, federal, freedom, congress, economic, future, budget, programs, spending, americans, together, growth, program, tonight, billion, percent, states.
Roosevelt	nation, people, government, democracy, spirit, national, states, united, progress, action, america, things, helped, purpose, without, leadership, millions, freedom, within, public.
Shakespeare (plays)	antony, caesar, hamlet, brutus, othello, macbeth, cleopatra, exeunt, cassio, cassius, desdemona, heaven, portia, juliet, octavius, cannot, horatio, enobarbus, nothing, father.
Shakespeare (sonnets)	beauty, though, praise, better, nothing, thoughts, beautys, thought, therefore, others, friend, gentle, things, heaven, whilst, change, making, summers, hearts, desire.
Yoda	training, trained, strong, master, apprentice, future, become, powerful, council, obi-wan, qui-gon, chancellor, discover, danger, senator, nothing, skywalker, grievous, knowledge, father.

These lists describe each author's distinctive vocabulary more completely. Notice that the difference between the two Shakespeare samples is much more obvious than before.

Random Sentence Generator

A *Markov chain* is a collection of states and a set of transition probabilities, where the probability transitioning to some state β is completely determined by the current state α. For each data set, we can create a simple Markov chain using the words of the text as the states. The transition probability $\mathbb{P}$ of transitioning from word α to word β depends on how many times β follows α in the text.

For example, suppose the word "William" is found three times as part of the names "William Shakespeare", "William Shatner", and "William Shakespeare" (again). Then if the Markov chain for this sample is in the state representing the word "William", it will transition to the state for "Shakespeare" with probability $\frac{2}{3}$ and to the state for "Shatner" with probability $\frac{1}{3}$.

Applying this process to our data sets and transitioning through the chains produces amusing results – can you tell which author 'wrote' each sentence?

1. Primarily, this pledge taken, I shall fail to speedy adoption.
2. For fear of many maiden virtue only is.
3. They concern, thank God, only know the extent to convert retreat into the overbalance of work, we apply social values more noble than mere possession of callous and a war against a conduct in emphasis and precious moral values; with a better use of locusts.
4. God of my unspeakable misery!—begone!

5. Be brought a storm, the night, or the angels name.
6. Dissuade one of the steep-up heavenly hill.
7. Twisted by my size, do not.

These 'sentences' are amusing[2], but nonsensical. We may conclude, unremarkably, that English must not be governed so much by the sequence of *words*, but by some other process. For this reason we avoid principal component analysis, which depends entirely on vocabulary and ignores other factors of writing style such as sentence structure.

Speech Tagging

Every natural language has structure and grammar, including a list of *parts of speech.* In English, common parts of speech include noun, verb, conjunction, and so on. The `nltk` module has tools for classifying the words of a sentence by their part of speech. This process is called *part-of-speech tagging* (POS tagging). The 46 different parts of speech in English that we will use, together with their `nltk` tag, are listed at the end of this paper.

To analyze each data set, we separate the file by sentences, split each sentence into words and punctuation, and tag them with their part-of-speech. Consider, for example, one of Yoda's first lines in *The Empire Strikes Back*:

I am wondering - why are you here?

The corresponding sequence of POS tags is:

PRP VBP VBG : WRB VBP PRP RB .

After reading and tagging each of our data sets, we are ready to begin with a more mathematical analysis of each author's style.

3 The Model

A discrete HMM has three components: a state transition matrix A, an observation matrix B, and an initial state vector π. Thus we distinguish a single HMM with the parameter $\lambda = (A, B, \pi)$. In a basic Markov model like the one used to generate the random sentences of the previous section, the states are specified. The states of an HMM are latent (hidden), and it is not possible a priori to know exactly what they will represent.

The Observation Space

Though we cannot specify the state space, we can (and must) define the observation space. The naïve approach would be to use English words as observations, as with the model for the random sentences. However, as there are an estimated 1.025 million words in the English language, that idea is infeasible (even if it were, it would probably not be very effective). Instead, we use the parts of speech as indicated by `nltk`, introduced in Section 2.

[2]Answer Key: 2 and 6, Shakespeare; 4 and 5, Poe; 1 and 3, Roosevelt; 7, Yoda.

Refining the Observation Space

A possible pitfall to using an HMM (and to Expectation Maximization in general) is if one of the observations in the state space never occurs in a training set of data. For example, the `nltk` tag UH is used for interjections. The Roosevelt, Sonnets, and Yoda data sets do not contain the UH tag, while the Poe and Shakespeare plays data sets do. Any sequence of observations that contains even a single UH tag cannot be classified with any of the authors that do not use UH. Consider for example, the sequence

$$\mathcal{O} = [\text{PRP VBP VBG : UH WRB VBP PRP RB .}],$$

which represents the text

“I am wondering - uh why are you here?”.

Clearly the sequence should be classified as coming from Yoda, but the UH tag makes it impossible for an HMM that is sufficiently trained on the Yoda data set to classify it as such. More precisely, the column-stochastic observation matrix B in the Yoda HMM will have a row of zeros in the row that corresponds to the UH tag. The algorithm to compute $\P(\mathcal{O} \mid \lambda)$ then suffers a `ZeroDivisionError`, and the classification process fails.

In our case, the tags

$ ” ( ) – FW LS POS SYM UH WP$ “

are missing from one or more of the data sets after speech tagging, so we remove these from the observation space to facilitate correct classification.

Start and End

In addition to the remaining 34 POS tags, we add two additional observations: the beginning of a sentence (denoted $tart), and the end of a sentence (denoted en&). This should provide a little additional structure, since most lines end with a sentence terminator and such. It should also make it easier to distinguish between Shakespeare plays and Shakespeare sonnets, since the plays have stage directions at the beginning of most of the lines.

Training the Models

To construct an HMM we must specify the number of hidden states. Since we don’t know beforehand what the hidden states represent, we train several different models for each author, each with a different number of states. Mathematically, we specify N, and find an appropriate $N \times N$ column-stochastic state transition matrix A and an $M \times N$ column-stochastic observation matrix B, where M is the number of observations ($M = 36$, in our case). We do this process for $N = 2, 3, \ldots, 15$, resulting in a total of 14 models for each author.

Before using the models to classify our data, it is instructive to examine the matrices themselves once the training algorithm has converged. Below is printed the transition matrix A and the observation matrix B for the 3-state model trained on the Roosevelt data set. Since each row of B corresponds to one of the possible observations (the parts of speech), by taking the maximum of each row we can separate the observations into N groups, indicated by the coloring of the text.

$$A = \begin{bmatrix} .01118459 & .0143557 & 1 \\ .60631363 & .50809421 & 0 \\ .38250178 & .4775501 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} .13024037 & 0 & 0 \\ .08575537 & .01574452 & 0 \\ .1297066 & 0 & 0 \\ .04945212 & .00101964 & 0 \\ .05272082 & .03952492 & 0 \\ 0 & .00611783 & .00708271 \\ 0 & .26081418 & 0 \\ 0 & 0 & .00769914 \\ .27833312 & .09598755 & 0 \\ 0 & .15335004 & .02917099 \\ .00000034 & .00610411 & .00160039 \\ 0 & .00303272 & 0 \\ .04101485 & .00383139 & .00457378 \\ 0 & .06297962 & .3721068 \\ 0 & .01783347 & .02630773 \\ .00003848 & .00060109 & .00689651 \\ 0 & .01886913 & .13828051 \\ 0 & .00346597 & 0 \\ 0 & .01757494 & .13112425 \\ 0 & .05545551 & 0 \\ .00267168 & .04877462 & .05027279 \\ 0 & .00433246 & 0 \\ 0 & .00259948 & 0 \\ 0 & .00243508 & .00185849 \\ .04004859 & .02771502 & 0 \\ .00000265 & .07274731 & .02094323 \\ .03435983 & .00015203 & .00565191 \\ 0 & .02166231 & 0 \\ 0 & .03263354 & .04546711 \\ .08872579 & .00380211 & 0 \\ .05335101 & .01301865 & 0 \\ .01205234 & .00179206 & .00950519 \\ 0 & .00380136 & .00782335 \\ .00152602 & .00222734 & 0 \\ 0 & 0 & .13363512 \end{bmatrix},$$

with initial state vector $\pi = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. The observations corresponding to each of the colored groups are listed below.

$start , . : CC IN MD TO VBD VBP VBZ WDT

DT JJ JJR JJS PDT PRP$ RBR RBS RP VB VBG WRB

CD EX NN NNP NNPS NNS PRP RB VBN WP en&

These results are encouraging, since many similar parts of speech are grouped together. All of the punctuation types are grouped in the red group, the adjectives in the green group, and the nouns in the blue group. This doesn't necessarily mean that the hidden states represent these groups of observations, but it does support the assumption that Roosevelt has a consistent style and voice.

As another example, consider the following groups obtained from the 8-state model trained on the Yoda data set.

NN NNP NNPS NNS

MD VBD VBP VBZ

EX JJR PRP RB VB WP

.

: CD DT JJ JJS PRP$ RBR RBS

en&

, CC IN PDT RP TO VBG VBN WDT WRB

$tart

Note that the sentence terminator, end of sentence, and start of sentence tags are completely isolated into their own groups. Once again all of the nouns are grouped together, though in this model the punctuation and adjectives are *not* all grouped.

Each of the author's models have logical, but distinct groupings like the ones listed above, indicating that this strategy for author identification may actually be effective.

4 Results

Nearest Neighbor Voting

We begin with an easy author-identification case. Suppose that we suspect that a mystery document had contributions from any of the 6 authors in our data set, and that the document was actually completely written by Roosevelt. For each $N = 2, 3, \ldots, 15$, we use the N-state model for each data set. After speech tagging the mystery document, we feed in each of its sentences as an observation sequence to the models. The author whose model has the highest probability[3] $\mathbb{P}(\mathcal{O} \mid \lambda)$ is selected as the label for that sentence. Initial results are positive:

Number of States	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Percent Accuracy	52	40	36	57	50	61	55	70	71	71	70	74	78	68

Though the accuracies are well above the expected value of guessing ($\frac{1}{6} \approx 17\%$), the results are a little poorer than hoped. This is probably because at this point, we have only considered each sentence of the mystery document in isolation. It is reasonable to assume that the author of a particular sentence has a high probability of being the author of the adjacent sentences in the document, since authors tend to write whole paragraphs or sections at a time.

To capture this idea, we use a kind of nearest neighbor voting scheme. Suppose we have three sentences s_1, s_2, and s_3 in sequence. For a given number of states N, we select the models of the potential authors as before. To classify the sentence s_2, we first run s_2 itself as an observation sequence through each of the models. The author whose model has the greatest probability is awarded 2 points. Then we concatenate s_1 and s_2 into a single observation sequence $s_1 s_2$, and run

[3]In practice we compute the log probability to avoid numerical underflow.

that sequence through the models. The winning author is awarded 1 point. Likewise, we award 1 point to the author that wins the classification for the joined sequence s_2s_3, and 1 point to the author that wins on the concatenation $s_1s_2s_3$. The author with the most points is declared the label for the sentence s_2. In the case of a tie, we select the author that was chosen for s_2 in isolation.

With this system, it is possible for the actual sentence s_2 to be misclassified initially (2 points to the wrong author), but to be classified correctly in the end if the sequences s_1s_2, s_2s_3, and $s_1s_2s_3$ are all classified correctly (3 points to the correct author).

See Figure 1 for a comparison of this method with the previous method.

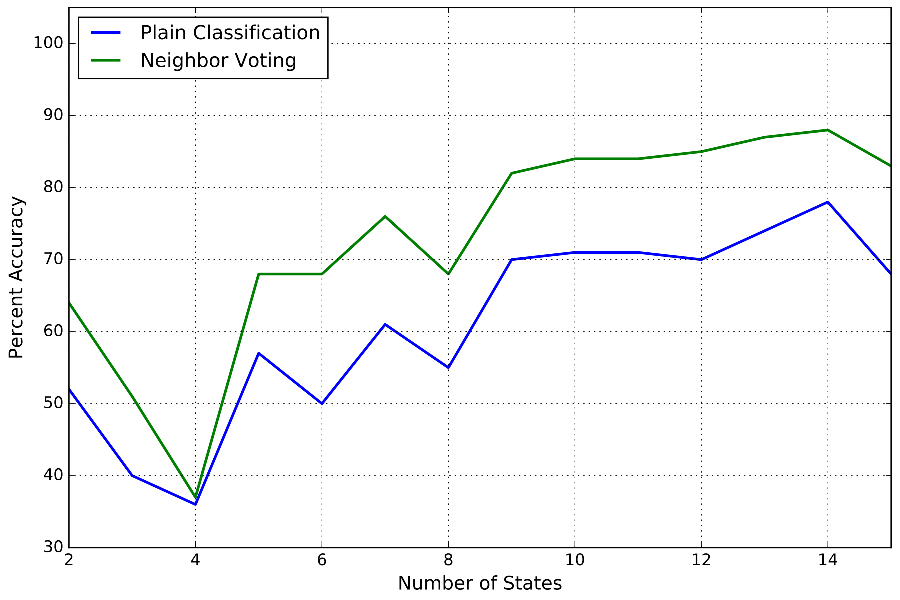


Figure 1: The voting method always outperforms the plain method, which is unsurprising since there is a single author for this particular target document.

There are, of course, some minor disadvantages to the voting scheme. If a particular mystery document had three authors, each writing every third sentence, the plain classification method might occasionally work better than the voting method because of the frequency in changes of authorship. Most serious collaborative works are not of this form, however, so we can safely ignore this concern. From here on out we will exclusively use the voting classification method.

Easy Tests

We now present a few test cases that should have clear outcomes. If the percent accuracies are not very high in these cases, then we should start worrying about this classification method.

Roosevelt v Yoda

Continuing the example in the previous section, suppose we have reason to believe that the authors of the Roosevelt mystery document can only be Roosevelt and Yoda. Narrowing the number of possible authors means comparing less models, so we expect the accuracy to rise. The results are displayed in Figure 2. As with the previous example, it appears that using models with 9-14 states give the best results. This isn't always the case, but it does suggest that having 9-14 states in the HMM classifiers may help us to identify Roosevelt in particular.

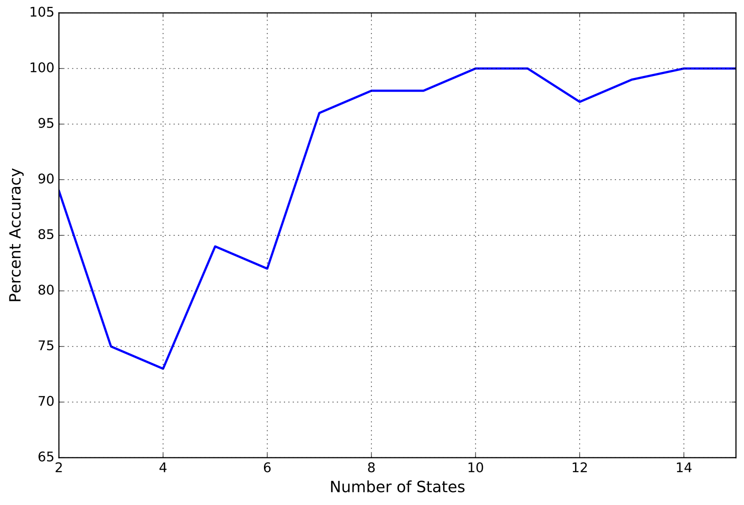


Figure 2: Roosevelt and Yoda have very different styles of speech, so it is unsurprising that the classification does so well.

Reagan v Poe v Yoda

To make things a little harder than the previous test, we suppose that there are three authors, and generate a mystery document that is actually a composition of writings by each author. The first third of the document will be by Reagan, the second third by Poe, and the last third by Yoda. Because these three authors are very distinct (one politician, one poet, and Yoda), we still expect good results. However, having 3 candidates instead of 2 will make it harder to get in the $90-100\%$ accuracy range. See Figure 3.

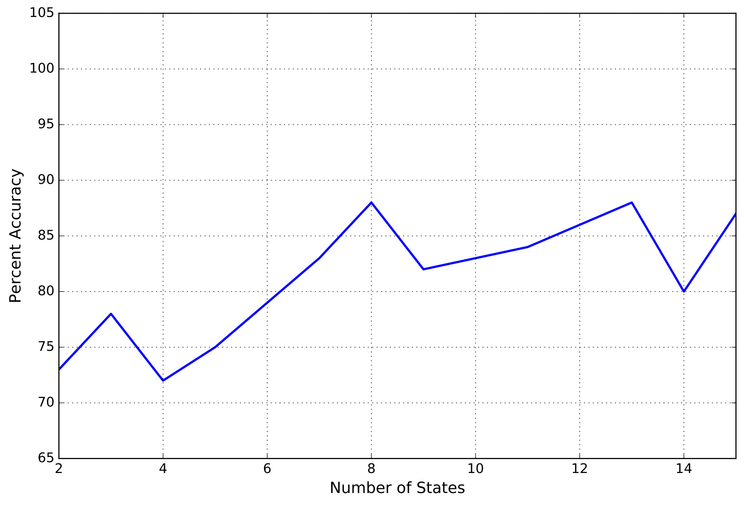


Figure 3: As expected the algorithm does fairly well, mostly getting $80-90\%$ accuracy.

Harder Tests

Yoda is a little too unique to constitute a real classification challenge, so we exclude his data set altogether for the next few tests. Instead we compare the politicians to each other, the poets to each other, and Shakespeare to himself. We'll use two different test documents for each matchup.

Reagan v Roosevelt

Since Reagan and Roosevelt are both politicians and their writing samples come from publicly televised events, we expect their styles to be similar. We test one file where Reagan authors the first half and Roosevelt the second, and another where they switch off every quarter of the way through (every 25 lines out of 100). Surprisingly, the algorithm still does fairly well! See Figure 4.

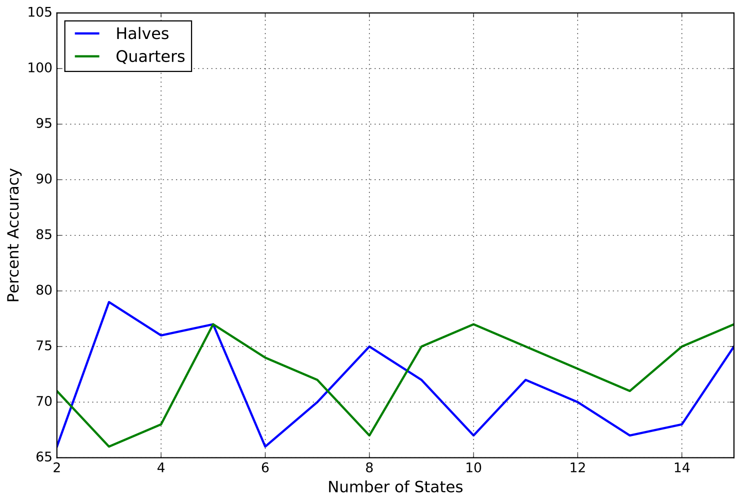
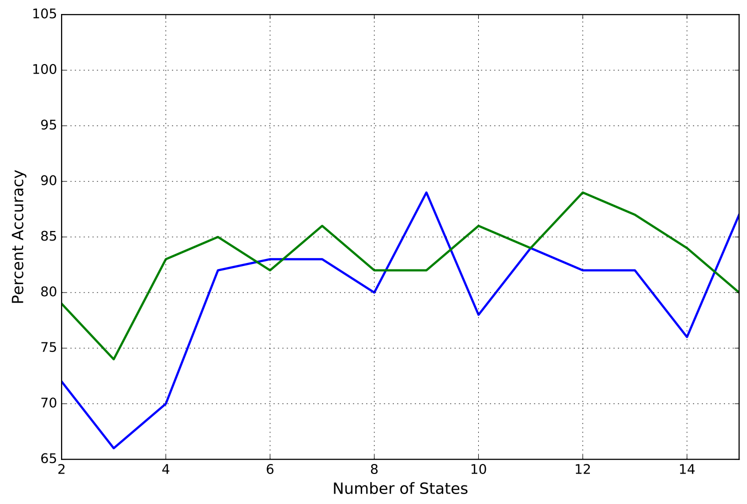


Figure 4: Even for this problem, the classification scores $65 - 80\%$ accuracy.

Poe v Shakespeare (Sonnets)

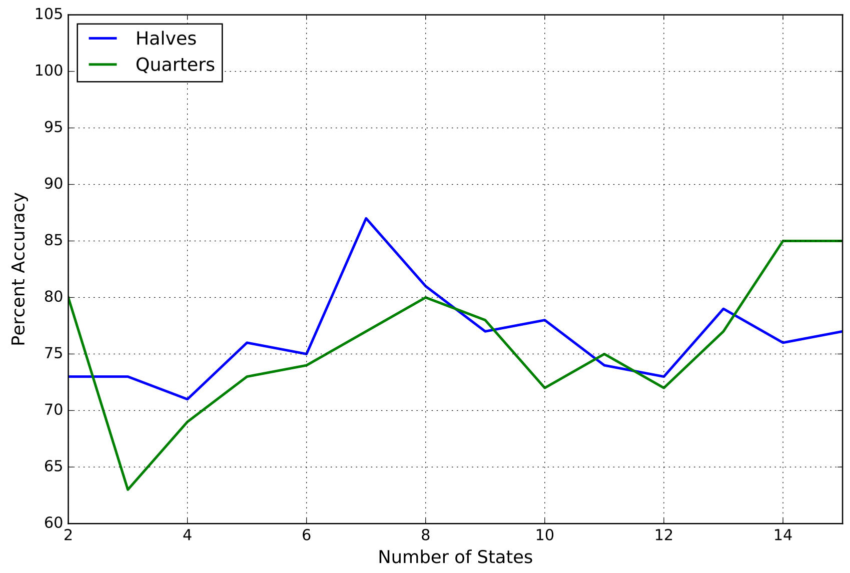
We similarly compare the two poets, with the same style of documents. The results displayed below are much better than the Reagan v Roosevelt case.



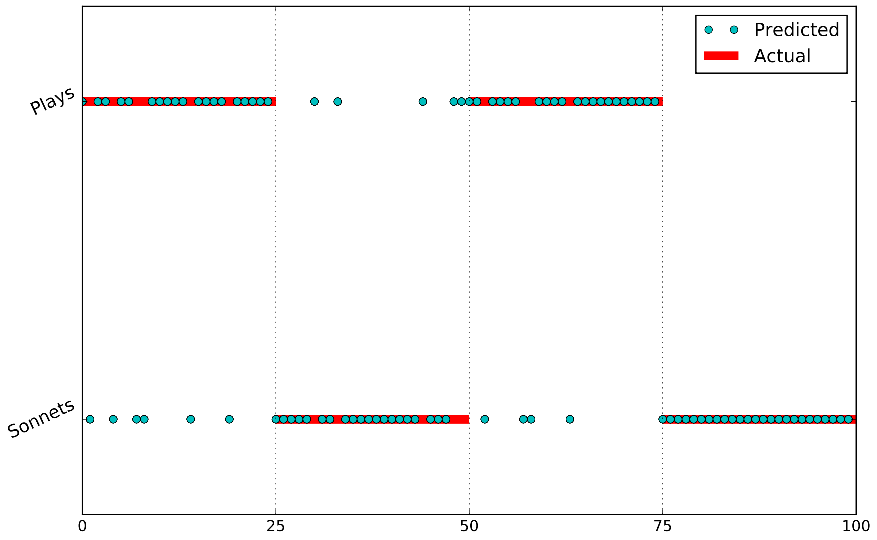
Since these accuracies are higher than those in the previous experiment, we conclude that Shakespeare's and Poe's styles are *less similar* than Roosevelt's and Reagan's styles.

Shakespeare v Shakespeare

As a final test, we compare Shakespear's Sonnets to his plays. This is an example of how this classification algorithm can be used to differentiate between *genres*, not just determine authorship. The results are surprisingly good.



So far we have displayed the accuracy of the models as a whole, but looking at the sequence of predicted labels compared to the actual labels can also provide useful information. With $N = 14$, the sequence of labels by sentence, together with the actual label, are shown below for the document that alternates between plays and sonnets every 25 lines.



Without the red line, it would still be possible to identify the passages or chunks of the document, with a fair amount of certainty, that were taken from the Sonnets and those that were taken from the plays.

5 Conclusion

The Hidden Markov Model approach to the author identification problem has few drawbacks, is generally very accurate, and doesn't take long to compute. Given enough training data, you will surely be able to prove Jack's claim false and claim your deserved recognition.

6 Appendix: POS Tags

On the next page.

Tag	Part of Speech
$	dollar
’ ’	closing quotation mark
(	opening parenthesis
)	closing parenthesis
,	comma
–	dash
.	sentence terminator
:	colon or ellipsis
CC	conjunction, coordinating
CD	numeral, cardinal
DT	determiner
EX	existential there
FW	foreign word
IN	preposition or conjunction, subordinating
JJ	adjective or numeral, ordinal
JJR	adjective, comparative
JJS	adjective, superlative
LS	list item marker
MD	modal auxiliary
NN	noun, common, singular or mass
NNP	noun, proper, singular
NNPS	noun, proper, plural
NNS	noun, common, plural
PDT	pre-determiner
POS	genitive marker
PRP	pronoun, personal
PRP$	pronoun, possessive
RB	adverb
RBR	adverb, comparative
RBS	adverb, superlative
RP	particle
SYM	symbol
TO	“to” as preposition or infinitive marker
UH	interjection
VB	verb, base form
VBD	verb, past tense
VBG	verb, present participle or gerund
VBN	verb, past participle
VBP	verb, present tense, not 3rd person singular
VBZ	verb, present tense, 3rd person singular
WDT	WH-determiner
WP	WH-pronoun
WP$	WH-pronoun, possessive
WRB	Wh-adverb
‘ ‘	opening quotation mark