EE155: Shakespeare Bot 5000

Akshta Athawale, Mannat Singh and Miguel Aroca-Ouellette

1. Overview

For the development of our Shakespeare Bot 5000 our team chose to take an iterative approach, wherein the first goal was functioning preprocessing, HMM training and Naïve Generation upon which incremental improvements were built. The incremental improvements first involved using visualization and qualitative sonnet analysis to determine an appropriate number of hidden states. Following this we added further preprocessing and more complex poem generation which used rhyme, meter and parts of sentence tags to generate a Shakespearian sonnet. Each step was tuned incrementally and qualitative analysis showed that each had an effect in generating a better poem.

We have three members in our team, Akshta Athawale, Mannat Singh and Miguel Aroca-Ouellette. Akshta was in charge of visualization and interpretation. Mannat was in charge of the unsupervised learning and HMM implementation. Miguel was in charge of preprocessing, poem generation and the additional goals. However, there was significant collaboration each task across the team, particularly with regards to debugging and qualitative sonnet analysis.

2. Unsupervised Learning

2.1 Preprocessing

Our team chose to tokenize the dataset by word, where a singular sequence would be represented by a line of a poem and each type of stanza (quatrain, volta and couplet) maintained a separate list of tokens. This approach was chosen as Shakespeare uses different intonation and language in each of his stanzas. The quatrain is the body of the poem and introduces the theme or story with lots of adjectives describing the subject of the poem. They maintain a consistent mood through, except for the volta, which usually represents a change in mood. This is grammatically demarcated by particularly strong adjectives and adverbs, as they have to provide a counterpoint in 4 lines to the 8 of the previous quatrains. By separating the volta from the other quatrains our team hoped to capture this shift in mood, or at least the shift in intensity. Similarly, although the couplet does not bring the same intensity as the volta, it does use different language from the rest of the poem as Shakespeare is bringing a close to the story he has presented. It usually starts with a conjunction (i.e. “or”, “but”, “yet”, “then”) and instead of posing the questions or accusations of the previous lines, it presents statements or final actions. Once again, by separating the couplet from the other stanzas, we hoped to capture his literary voice.

Originally, we had removed all punctuation when tokenizing words but found that we were often left with nonsensical words such as “consumst” instead of “consum’st”. We quickly realized that Shakespeare often uses apostrophes as an alternate spelling to words or to omit syllables. Similarly, he used hyphens to alter the meaning of combined words. So, in the end when tokenizing the data, all punctuation was stripped except for apostrophes “ ‘ “ and hyphens “-“.This also allowed for words implying possession and plural words to stay separate, which we expected to be beneficial as such words appear in different contexts and with different meaning. Some examples of the problems fixed by leaving in the apostrophes are “consum’st” and “murd’rous”. Hyphens fixed combinations such as “all-

eating", which means self consuming, versus "all eating" means everyone eats. Other punctuation was added back during poem generation.

During preprocessing, we also used the line endings and the known rhyming scheme (*abab cdcd efef gg*) to create a rhyming dictionary which was later used by our poem generation algorithm to ensure rhymes at the end of the lines.

2.2 Unsupervised Learning

In order to perform unsupervised learning, we used the Baum-Welch algorithm as laid out in the lecture slides and HMM notes for multiple training sequences. The stopping condition was the convergence of the Frobenius norms of both the state transition and the observation matrices, which we checked by looking at whether the fractional change in norms was below a threshold for both the matrices. The number of hidden states was chosen qualitatively by inspecting the sonnets, as well as using the hidden state visualizations discussed below.

With our first version of code, it used to take ~ 1 hour to iterate over the set of sequences for one EM step with 5 hidden states. We realised that in order to tweak and test a range of model parameters and pre-processing techniques within the time we had, we needed to improve upon the HMM training time. A few optimisations were made –

- To calculate the marginal probabilities, the denominators don't need to be re-calculated each time and can be stored in memory
- While recalculating the O matrix for a sequence, iteration over all the words is unnecessary, only words which occur in the sequence will have their numerators updated, and the denominators will remain the same for each word for a given state – this was the most time consuming step without the optimisation

These reduced the runtime tremendously, resulting in the time for one EM step dropping down to ~ 2 minutes (for 5 hidden states), allowing us to train the HMM quite frequently with different inputs and parameters.

Qualitative analysis showed that increasing the number of states allowed for better sentence structure, however this came with diminishing returns in creativity and computation time. We found that a good balance between these two factors was using 30 hidden states, as each state captured enough information from the poems to generate grammatically probable sequences while still generating varied and creative poem lines. See Section 4 for a sonnet generated with 30 hidden states and the Appendix for sonnets generated with other number of hidden states.

3. Visualization and Interpretation

3.1 5 Hidden States

To try and interpret what our model has learned, we started with looking at the top 10 words of each of the 5 hidden states, but as the probabilities were not normalized we ended up with the most common words for most of the states. After normalizing the probabilities by frequency of word occurrence, we were able to observe groups of words for each state. Below are the top 10 most common words for each of the 5 hidden states:

State 1: *bold, minutes, won, outlive, invocate, happies, climbed, sounds, unions, equipage*
State 2: *lovers', jade, shines, breathes, cast, owes, ignorance, drops, i'll, blunt*
State 3: *stone, whether, slave, consum'st, wife, stick'st, imprisoned, race, convertest, silvered*
State 4: *erst, profound, threw, grave, fall, doubting, cools, tyranny, 'greeing, crow,*
State 5: *makes, slandering, constancy, sacred, compile, blamed, swart-complexioned, ear, wand'rest, widow's.*

We analyzed these words using their position in a sentence, stress pattern, number of syllables, the sentiment, and part of speech. Unfortunately, the first three approaches did not yield satisfactory results, but based on the sentiment of the words we were able to categorized these states into the following sentiments.

State 1: Success (bold, won, climbed, happies, outlive),
State 2: Love (lovers', jade, shines, breathes),
State 3: Confinement or being forced (imprisoned, slave, stone, stick'st, convertest, race*)*
State 4: Negative or dangerous (threw, grave, fall, doubting, cools, tyranny, crow)
State 5: Positive (makes, constancy, sacred, compile, ear)

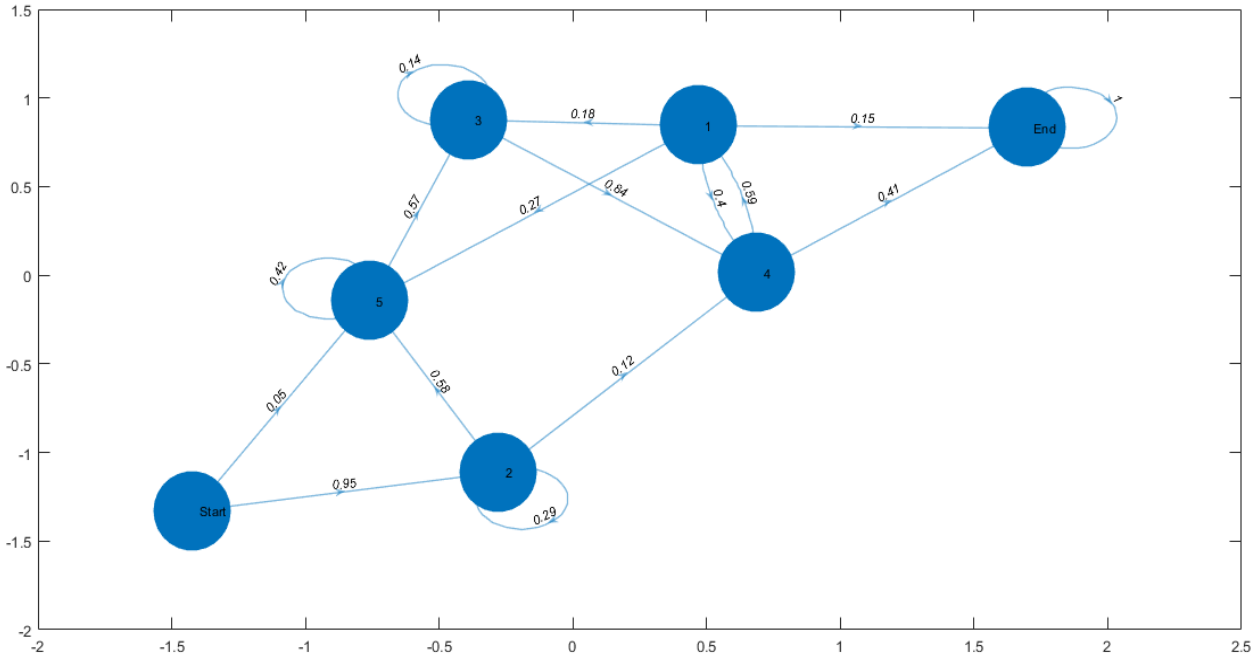
To get a better understanding of these groupings we then looked at parts of speech (POS- generated by each state. Specifically, we looked at the probability of every state emitting a particular part of speech. Noun, Verb and Adjective were the most frequent part of speech in most of the states, with Noun dominating in a lot of the states. This was not very informative because the probabilities of occurrence of these three POS tags were similar. To solve this, we instead looked at the frequency normalized probabilities of a POS tag occurring at each state. This yielded the following as the top POS tags for each state.

State 1: Verb, Noun, Adjective
State 2: Noun, Verb, Adjective
State 3: Adjective, Noun, Determiner
State 4: Adverb, Noun, Adjective
State 5: Noun, Preposition, Adjective

In addition, we analyzed parts of speech of the top 10 words of each state; these are listed below. Using both these information sets we concluded that *State 1* is most likely to be a *verb*, *State 2* a *Noun*, *State 3* an *Adjective*, *State 4* an *Adverb* and t*Sate 5* a *Noun* again.

State 1: *Adjective, Noun, Verb, Verb, Verb, Verb, Verb, Noun, Noun, Noun*
State 2: *Noun, Noun, Noun, Noun, Verb, Verb, Noun, Noun, Noun, Noun*
State 3: *Noun, Preposition, Verb, Verb, Noun, Verb, Verb, Noun, Noun, Verb*
State 4: *Adverb, Adjective, Verb, Noun, Noun, Noun, Verb, Adjective, Verb, Noun*
State 5: *Verb, Verb, Noun, Verb, Noun, Verb, Adjective, Noun, Adjective, Noun*

For analysing what transition states capture about the data, we plotted the transition matrix as a state diagram.



This yields the following interpretations.

- The first transition is almost always to state 2, which means the first word is generated in a specific fashion as one would expect.
- We only reach the end state from state 4 or 1 (with a low probability). This is also expected, since we only want to end a sentence only when certain conditions are met.
- We go from state 3 to state 4 with a very high probability (85%). This models word pairs – certain set of words are very likely to be followed by other words. Perhaps adjective and nouns.
- Every state only transitions to a few other states, which means there is some flow to the sentence.

3.2 30 Hidden states

We attempted a similar analysis on our HMM with 30 states. We noticed that interpreting lower number of states is easier but as we go higher, each state capture a more complex combination of attributes, so it is much harder to visualise or interpret each state. Below are the top 10 words from the 30 states

State 1: *harsh, dye, ambush, perceiv'st, measured, increase, sessions, seldom, striving, flower,*
State 2: *stone, general, herald, kingly, believed, drinks, maketh, feeble, starved, drudge,*
State 3: *numbers, ruin, necessary, took, delight, freedom, ashes, full, shun, pilgrimage,*
State 4: *forbidden, soundless, torture, half, kind-hearted, silent, store, extreme, up-locked, bears,*
State 5: *saved, fearing, office, works, race, enlighten, possession, brave, greet, expired,*
State 6: *print, compare, tattered, becomes, tires, mistress, hill, silver, what's, legacy,*
State 7: *slow, crowned, whereupon, abuses, thunder, face, bootless, ear, situation, belongs,*
State 8: *acceptable, suff'ring, gross, ceremony, mournful, welcome, where-through, intend, hope, trees,*

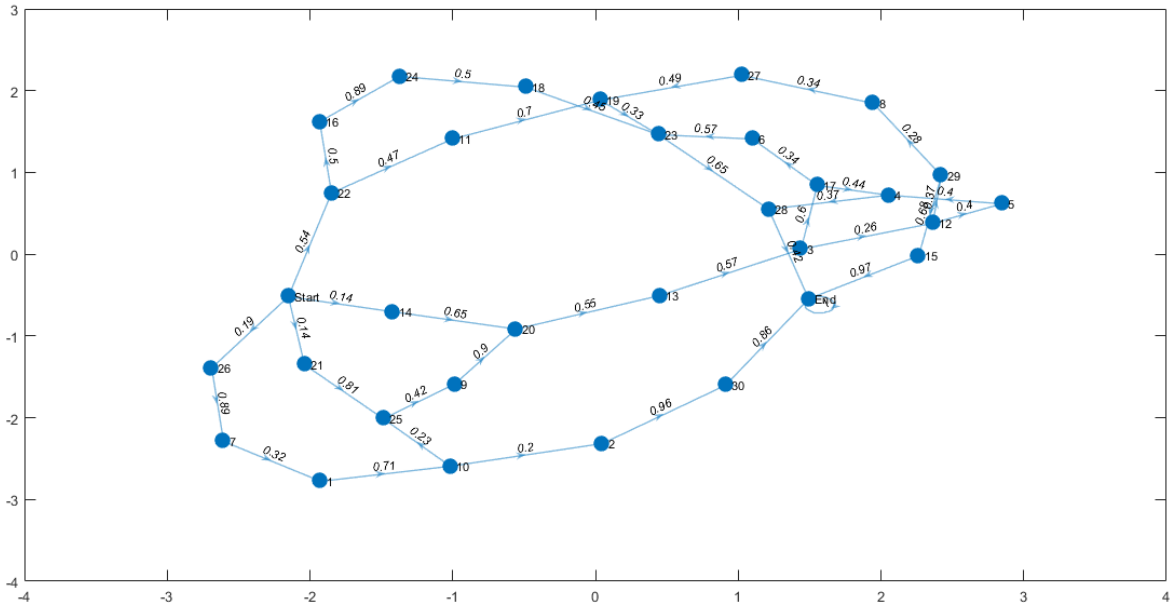
State 9: *executor, pierced, vaunt, self's, wisdom, married, under, bold, light, score,*
State 10: *anticipate, distilled, passed, god, abysm, lo, simplicity, blunting, victories, single,*
State 11: *ne'er, uneared, mine, slandering, pleasant, long-lived, mixed, afford, heat, five,*
State 12: *certain, throw, impanelled, sweetest, composed, lives, false, writ, hues, tyranny,*
State 13: *means, foot, falsehood, raven, banquet, break, temptation, outlive, yea, constancy,*
State 14: *double-vantage, verses, evident, mud, bristly, prize, defendant, a, tallies, hied,*
State 15: *receiv'st, survive, perfection, me, wherever, shows, embassy, unrest, monuments, pursuit,*
State 16: *part, afloat, once, stopped, self-loving, point, lets, lengths, grievances, ransom,*
State 17: *happy, fools, painting, climbed, unfolding, despair, eye, ruinate, wane, crow,*
State 18: *prognosticate, whereof, accessary, been, slave, heart's, scythe, sums, our, tempting,*
State 19: *spend, permit, 'not, go, straying, his, i, statute, commanded, curls,*
State 20: *pluck, sea, help, beloved, chance, o'ertake, for, universe, died, render,*
State 21: *dearer, adding, length, anew, thorns, children's, mourn, titles, will, shamed,*
State 22: *lov'st, say, supposed, no, pale, mansion, teach, graces, special-blest, poor,*
State 23: *singleness, beguile, flowers, speak, miracle, defence, discased, tear, uphold, trespass,*
State 24: *their, extremity, rider, bar, shadow's, astronomy, rid, frailties, promise, waiting,*
State 25: *provoke, cures, hot, princes, good, tanned, abundant, winds, foul, perfumed,*
State 26: *influence, favourites, matcheth, cries, wrinkle, already, sort, vanished, love', deepest,*
State 27: *bearer, staineth, wand'rest, gladly, 'thou, hiding, morning, dare, chips, glutton,*
State 28: *rise, appearance, small, field, stirred, word, issueless, kings, losses, repent,*
State 29: *called, ill, get, blanks, wink, ah, shop, seem, nature's, dove,*
State 30: *'greeing, unjust, knowledge, like, carcanet, brass, brav'ry, scorn, cools, pointing,*

As is apparent, it is particularly difficult to decipher the sentiment of each state. We have tried interpreting some of the states below.

State 1: *quantity or comparison (harsh, measured, increase, seldom)*
State 4: *Tragic (forbidden, soundless, torture, half, silent, store, extreme, bears)*
State 8: *death (suff'ring, gross, ceremony, mournful, trees)*
State 20: *love (sea, help, beloved, chance, universe)*

Similarly, it was more difficult to analyse the part of speech for each state, as no part of speech was clearly dominant in any of the states. However, in general Noun, Verb and Adjectives were still the most common parts of speech

Below is the state transition matrix plotted as a state diagram.



Note: We have plotted only part of the matrix. We have removed the edges with zero or very low probabilities.

This diagram allows for certain insights on the states, particularly:

- Same as the case with 5 hidden states, start goes to one of the states (22) with a very high probability, which means the first word is generated in a specific fashion
- End can only be reached from 3 out of 30 states, viz. 15, 28 and 30
- There are some states in this case as well which mostly occur in pair, viz. 16 → 24 with 89%, 2 → 30 with 96%, 26→ 7 with 89%
- And lastly, every state transitions to only 2 or 3 other states with a significant probability. So, no abrupt jumps in this case either

Thus overall, although the 30 hidden state case is much more complex and difficult to interpret, it does yield similar properties to the 5 hidden state case.

4. Poem Generation

To generate our poems, we use three trained HMMs, one for each stanza type. Each stanza then used its token library to individually generate the appropriate number of lines for that stanza. The stanzas were then combined in the correct order to generate our Shakesperian sonnet. The first letter of each line was capitalized and punctuation at the end of the line was added randomly; with a high probability

simply a comma was added, and with a small probability an exclamation mark, question mark or semicolon was added. Using this naïve generation method and 30 hidden states Sonnet 1 was generated and is shown below. In the Appendix, Sonnets A1-A5 demonstrate the effect of different number of hidden states on naïve generation.

Sonnet 1

Which upon thinking for their head, *(1)*
But seeing sauces patience do war,
And face thy pipe thy if my function swear therefore are temptation invited,
But dost and where stain all strange,
And that those things wilt thee dreams true, *(5)*
Your very I shall think begins asked than and is needing clock to bright,
When lovely reproach harvest defaced;
That so admitted lame do altered days,
He newer their thoughts of suited point a him proof fear brag that they thee,
Against what of gracious the great, *(10)*
And I walks truly by thy lines thine air,
That I and doth at of such yet nor to must everywhere,
And thy thy you that thy store.
For so need besides thou knows mind.

Although mostly nonsensical, sonnet 1 has a few lines which appear to maintain Shakespeare's voice and are intelligible, such as lines 1, 5,8 and 11. Overall it also has a Shakespearian feel due to the word choice and the overall sentence flow of each line. Particularly, we note that many of the lines (1,2,3,4,7,10,13,14) start with conjunctions, which is one of the grammatical characteristics of Shakespeare's sonnet writing. The HMM model was able to capture some of the aspects of Shakespeare's line structure as the hidden states represented a loose grouping of parts of speech tags. This becomes particularly clear in the starting conjunction characteristic. Because this part of speech always occurs at the same place in the sentence the HMM is able to accurately capture and replicate this characteristic, as is clear in our generated sonnet. This relationship is not as clear cut within the sentence where parts of speech, rhyme and meter are much more varied for each word. At the end of sentences HMMs were able to capture the fact that Shakespeare (and sentences in general) rarely ends a line with a preposition or article. This is most likely because states which have a high probability of generating preposition or articles only have a small probability of transitioning to the end state. Both of these observations correspond with our result from the HMM visualizations.

Unfortunately, the poem also lacks some of the key characteristics which defines a Shakespearian sonnet. First of all, it does not match the rhyming scheme at all. However, this is not particularly surprising since naïve generation does not take into account rhymes and the probability of randomly generating two end words which rhyme is small. The sonnet also doesn't have proper syllable counts with lines ranging from 19 syllables (line 3) down to 7 syllables (line 13). Since the naïve generation line stopping condition is simply based on transitioning to the end state and we have assumed the Markov property, this large variety in line lengths is not surprising. Similarly, the sonnet fails to capture the meter of Shakespearian sonnets; line 5 and line 13 are almost entirely composed of stressed syllables.

Comparing this sonnet to Sonnets A1-A5 which were generated with a different number of hidden states we see that the main difference is in the quality of the sentences. Sonnet A5 (50 Hidden States) yielded a similar number of intelligible sentences as using 30 hidden states, while Sonnet A1 has no intelligible sentences. This corresponds with the expectation that a larger number of hidden states better captures sentence structure, but this increase in hidden states is met with diminishing returns and exponentially longer training times. We also note that the increase in hidden states has little or no effect on the quality of the rhyming and meter within the sonnets.

5. Additional Goals

5.1 Rhyme

In order to add rhyming to our Shakespearian sonnets, during preprocessing we generated a rhyming dictionary. To allow for rhymes to be the seed word of each line, every poem line was reversed before being processed by the HMM training algorithm, this allowed us to model reverse transition probabilities (reading sentence right to left) instead of forward transition probabilities (reading sentence left to right). Using these new state transition and observation matrices we could then generate each line of the poem backwards. This was done by randomly selecting a word with a rhyme in our rhyming dictionary and using that as the seed for the generation of our poem line. If the line, was a second rhyming line (i.e. the second *a* rhyme line), then the seed was selected from one of the rhymes of its previous rhyming line (first *a* rhyme line). This was done for each line pair of the rhyming scheme.

This ended up working very well and yielded much nicer sonnets than the non-rhyming approach. These sonnets also matched much more closely the sonnets written by Shakespeare since the rhyming scheme was an important characteristic of his approach. The only limitation this imposed is that the final word of each line (the seed word in the line generation) had to come from the rhyming dictionary. We felt that although this was a little limiting, since the dictionaries were so large it did not limit the creativity of our Shakespeare Bot too heavily. In the future it would be beneficial to generate rhymes for many of the words in the poems, perhaps from an external rhyming dictionary source. This would allow for a larger variety in the rhyming schemes and greater variety in each line since there are more possible seed words. See Sonnet 2 below for an example of a sonnet generated once we added rhyming.

Sonnet 2

The are is be from fair seen brains eyes, *(1)*
Though lest shall o'erpressed said,
Which where moving spies,
My wide a shall the morning hath clock must prime an allayed;
To as farther whether o'er refuse streams womb one love, *(5)*
Stealing did to-day the embassage,
As since frost and doth above,
For my sins which simple doth vassalage,
But die pursuit guess day pass and rank back do to knife,
Mine shows is be, *(10)*
And when may thy farther most orphans life,
Though love for delights deepest his by thee did see,
As suborned do art of with old,
I this yet doth death be told.

As intended, sonnet 2 now matches the Shakespearian sonnet rhyming scheme. However, it does not solve the meter and syllable length problem encountered with the naïve generation; line 4 has 14 syllables and line 12 has almost entirely stressed syllables. These issues are addressed below.

5.2 Meter

As mentioned above, one of issues we noticed with the naïve poem generation was that without control over the meter, the generated sonnets did not match particularly well the linguistic flow of Shakespeare's sonnets. In addition, often our lines were simply too long and wordy as our line ending condition was based on word count or transition to an end state.

In order to remedy these problems, we decided to incorporate syllable counting. Syllable counting allowed us to generate sentences which roughly 10 syllables (as a Shakespearian sonnet should have), by using the syllable count as a new ending condition for our sonnet line generation. Syllable counting was done using the python *NLTK* [1] library, as well as *pyHyphen* [2] for words that are not present in the *NLTK* library. Although the syllable counting is not 100% accurate, its approximations were sufficient for our purpose.

Adding syllable counting also allowed us to keep track of the current syllable stress within a poem when parsing the training poem lines. Using the *NLTK* library, we were also able to provide syllable stress information for certain words. Combining the syllable counting stress estimate as well as the *NLTK* stress estimate allowed us to show the multiple possible syllable stresses that each word could have; for example, content as in happy, or content as in material, both have different stressed syllables.

When generating each line of our sonnets, we ensured that each subsequent word had the correct starting (or ending when generating backwards) syllable stress in order to maintain the expected Shakespearian meter. This was done by pruning all the words which did not have the appropriate ending or starting stress syllable and re-normalizing the corresponding column of the observation matrix over this new, controlled distribution.

This yielded particularly large advantages in imitating Shakespeare's style. Although it was slightly limiting in terms of creativity, since we were on average only pruning ~50% of each word from the observation distribution we still had a large subset of words to choose from. This was in some sense actually a good feature as in guaranteed variety of words within a sentence and avoided repeating highly probably words. See Sonnet 3 below for an example of a poem generated using both rhyme and meter.

Sonnet 3

Which brief were kingdom my thy I is smell!	*[10]*	*<0,1,0,1,0,1,1,1,0,1>*
Bad picture's thousand usest and your thy deem,	*[11]*	*<0,1,0,1,0,1,0,0,1,0,1>*
If praise the image for my ever tell,	*[10]*	*<0,1,0,1,0,0,1,1,0,1>*
Persuade in are muse proud-pied the seem,	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Them purple green my once forget in keep;	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Revenge alack but with feathers to your eye,	*[11]*	*<0,1,0,1,0,1,0,1,0,0,1>*
Eat harder how if poor to well to weep.	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Ay laughed still thinking not would with can fly,	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*

New but to of addition mistaking,	*[10]*	*<1,0,1,0,1,1,0,1,0,1>*
My graciously is health familiar be,	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Thy graciously against had thy making,	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Nor horse cool bring for reason you poor see,	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*
Skill by ne'er leads o put then than and end,	*[10]*	*<1,0,1,0,1,1,0,1,0,1>*
High spent and swear but for what dear a friend.	*[10]*	*<0,1,0,1,0,1,0,1,0,1>*

Using meter and syllable count consistently generated better poems, as shown by Sonnet 3. The new restrictions do not appear to have affected the intelligibility of the poem and have aided in generating a more accurate Shakespearian sonnet. We've manually annotated the number of syllables by each line as [⋯] and the stress of each syllable as <⋯> where 0 represents unstressed and 1 represents stressed. As we can see the syllable count is now almost exact to the characteristic 10 syllables. The few differences arise due to words which are not in the *NLTK* syllable dataset and are not properly parsed by *pyHyphen*; for example, usest in line 2 is two syllables, while our automated algorithm labelled it as one. Similarly, the stressed/unstressed syllable pattern is thrown off by words that are not in the dictionary or were not properly labelled by our automated algorithm. Unfortunately, the mislabelling of syllable stress can cause a cascade effect as each word is generated based on the previous words stress. A good example of this in the above poem is the word "o" in line 13, which is mislabelled as being high stress while it should be low stress. This mislabelling cascades and the next 4 words are generated with the wrong stress pattern (recall that line generation is done from right to left). However, overall, taking into account the meter of the poem was very beneficial to replicating the Shakespearian sonnet style.

5.3 Part Of Speech (POS) Restraints

We noticed that one of the grammatical difficulties which we were sometimes running into with the sonnet was repeated part of speech (POS) tags. For example, in the generated sentence "Level *inspire convert* device perish", both "inspire" and "convert" are verbs and it is grammatically quite rare for this to happen; one exception would be the sentence "He *began working* on his project". As such, as a general rule we decided to enforce that POS tags of one type could not be repeated sequentially; i.e. a verb could not follow a verb. We also enforced that certain tags could *only* occur sequentially (i.e. an adverb must lead a verb, a pronoun must lead a noun and a preposition must lead either a noun or pronoun) and that certain tags could never occur sequentially (i.e. a preposition should never lead a verb). POS tagging was done using the *NLTK* library and the data was then stored in a dictionary for quick retrieval. Many of the tags provided by the *NLTK* library were too descriptive for our purpose, so they were grouped into broader tags; i.e. "Determiner", "Predeterminer" and "Wh-determiner" where all grouped under "Determiner". See Sonnet 4 below for an example of a sonnet generated using rhyming, meter and POS retraints.

Sonnet 4

Death o and second mountain may no eye;
Upon is dignifies only anew,
Cost and dost do brand and to history,
Help kiss deserv'st thinks to keep thy hue,
Cheered worth's contend then did which fresh self is,
But which of well-contented endured,
Fresh pity more wink root to heinous this;

Breathed such as put and time were much assured,
In dumb the others sweet and lives wouldst bright,
With gazers thee but triumphant with expressed,
She health growing and words in eyes of night,
Them prouder sweetly gaze or out unrest!
Which untrue silent fool is dear love thee,
Love beauty after an by be the me.

It's much harder to evaluate the effect of the POS tagging restrictions, however it appears that in general this avoids some common grammatical mistakes which previous versions of our generation algorithm were encountering. For example, in line 10 of Sonnet 2 we have the double verb "is be" and in line 13 we have the double preposition "of with". Overall, this POS restriction approach yielded favorable results and qualitatively generated grammatically correct sentences slightly often than before. In addition, it had the added benefit of adding more variety within the poems as highly probably word sequences were now broken by the POS tag of the current line's randomly generated seed word. Since there are so many words for each POS tag this did not noticeably affect the creativity of our poem generation algorithm.

5.4 Additional Texts

One of the things we learnt from the Kaggle project was that having a larger dataset is often beneficial. There was not a specific problem we were trying to solve with this addition to our Shakespeare Bot, but we felt that perhaps the larger dataset and rhyming dictionary would be beneficial in adding creativity to our poetry generation and counteract some of the limitations imposed by rhyming and meter.

We added the Edmund Spenser dataset as well as sonnets from some of Shakespeare's other contemporaries which abided by the Shakespearian sonnet structure; these included John Keats, John Clare and Robert Frost. The result of our expanded sonnet dataset can be found below in Sonnets 5 and 6 below.

Sonnet 5

My new-found 'twixt that have beauty's apply,
They rules myself my gifts and 'tis this mask,
And brand but pride for love have mollify,
Full plaints thy being men the dust of task,
Huge me breath they thou golden silent made!
Would bright contentment on a night of youth,
Led inward broke to pleasure to my shade,
Hill brief awhile a thou quick with my truth,
Comptroll chopt but it such be mouth to breast,
Has where kindle dangerous thou delight,
We whilst when critic leisure ghastly chest,
Me nor those other prouder my shall white,
Doubt to images they upon her time,
And thee who self your lov'st proud will prime.

Sonnet 6

Thy monument of each from middle speak,
Depraves times o'erworn of came to wrong,
Hill but have I have can five filled or break,
Do many shine of bird from thee to tongue,
Shall whose should hours have ah see her light,
But glass to is his weary and of bent,
Friend's nor i'll finishing hath catch self quite?
Though dear subject a greatest discontent?
Cruelty speaking nothing sad her sever,
Fled clock fled eloquence let the heaven,
Would annexed pride the ripening persever,
Conspire lest whate'er be to even,
Her give to stones thy beauty and my heart,
She grace and why but own your leach my part.

Sonnet 5 and 6 do not appear to be distinctly better than Sonnet 4, however as expected the variety of words is greater and the algorithm now generates consistently more interesting sonnets. This arguable helped counteract the restrictions on rhyme, meter and POS previously applied.

6. Conclusion

In conclusion, our team was successful in implementing a Shakespeare Bot which produced Shakespearian sonnets of a reasonable quality. By visualizing the hidden states and analyzing both quantitatively and qualitatively their effect on poem generation we chose to use 30 hidden states for our final Hidden Markov Model. We then took several steps to improve upon the basic HMM implementation and naïve poem generation, and each modification had a visibly positive effect on the quality of the poems. Rhyme scheme and meter were the most important modifications we added in replicating the Shakespearian style, however controlling for POS was the most important in improving the reliability of generating grammatically correct sentences.

7. References

[1] NLTK Project, "Natural Language Toolkit," 03 03 2016. [Online]. Available: http://www.nltk.org/. [Accessed 10 03 2016].

[2] D. Leo, "PyHyphen 2.0.5," 2014. [Online]. Available: https://pypi.python.org/pypi/PyHyphen. [Accessed 10 03 2016].

8. Appendix

Sonnet A1 (5 Hidden States)

From else alone know,
Time's yield my call of to esteemed wantonness thou without my jewels you more,
This first crawls gentle with are stick'st wisdom excellence releasing,

Worth hate tables why thy so a speed thy know minds,
Art betwixt in thinly in be,
And in is self decrees will the thee the me and form self some so,
That betraying return fear with forgoing ear.
When it extreme turns me within trust sound me,
From proof a fair?
O those it praise thou all thither tie thine of love.
Cloud something as if new sing lovers rhetoric kind,
That every my new self in lost sweet doth but kind life rank bright,
No things so thy you takes rich my told?
Thy this from own by women's king have to to begin.

Sonnet A2 (15 Hidden States)

How that mine define,
Growth guilty the stop golden love I bring consum'st each bear heaven I is he a old fair time;
While fitted,
But in hand end,
That sweet late away cries,
Or fear on man although for be function his increase,
Numbers so I laid helen's with not grant motion fortune sweet-seasoned yet fearfully with do not,
And doth too are of few me beauty his thing white use?
To a to my just war works fair pay but the eye thy aye look drawn doth she in lawful it first self fair cured,
And time's judgement despise,
From lest kind my on say die never like wardrobe it thy,
Shall all-oblivious my so!
The thou want truth matter,
Where not winter thence virtue assure with thus as tongues heats moan.

Sonnet A3 (20 Hidden States)

Will therefore I the ornament as for of breast fading own excel,
Th' didst ripe way,
The form a be store,
That merit wires eye riches plead thee deeds,
For the hand of much be thee!
And purpose bide,
And thy these a have but that sight,
Ah those beauty to to live,
No wretch with cured,
Though black reckon obsequious all did shall well your with wrinkles beauty my all every me well of fits;
But locked on thence daily why hand wondrous quest youth,
But where that the rarities ornament,
Thrice in thou slander yet my I be so,
To him sleep is a men doth right the am or and this wealth I hearsay longer.

Sonnet A4 (25 Hidden States)

In hath abundance mistress make woman by december's once counterfeit,
My used be captain two would by deserv'st to treason,
Yet nor be frame of think view,
Fairing with sad lace leave confounds deeds are my holds should be words?
Look sweetest wood's of single forgotten!
Since services write are;
Thing of tie me masked and esteem,
If a which the wrinkles above,
So my smell seeing either return I will graces more,
But canst who to bud ears will own,
The thought of all taught stol'n world care,
Sets all now building more,
I thine be me,
When you outright with blot.

Sonnet A5 (50 Hidden States)

Which towers me hath dwell,
Who lusty predict think that dull war travel that my i'll black like maturity pilgrimage,
What's oft why a when great altered springs to rider wand'ring impart,
Profitless the basest are that the those thou self-substantial dulness,
Whose which olives from them with prayers ill,
The oft thy are oaths prizing misplaced,
Music have to not all eternity away,
Distilled heavy diseased night which some preserve gentle flower,
But do false words new,
Shall they thee rhyme,
But thy must of me are tell?
Yet what foul your of grow grief,
That eye mine thy ne'er thee beauty love love of thee,
'tis thy must friend go belong.