Logbook

Assignment 2: Bad Poets Society

Roman Guérin, Marinus van den Oever

Initial step

Firstly, we started to explore the Rita library. We experimented with *'The Rime of The Ancient Mariner'* for our input file. The long length of the poem allowed the program to generate lots of sentences that were based on a general theme. We looked at different forms of poetry as a guiding framework to display the generated sentences. We were influenced by Shakespeare, and a sonnet looked like something we wanted to get into. A sonnet has a fixed structure of 14 lines and uses an ABABCDCDEFEFGG rhyme scheme. The program generated grammatically correct sentences, but contained many duplicates, see figure 1.

I fear thy skinny hand, "quoth he.
Then darted to the land of mist and snow.
That agony returns; And then it left me free.
They stood as signals to the ship to go.
Still hid in mist, a shape, I trow!
The Moon was high; The dead men stood together.
But ere my living life returned, I trow!
Are those her sails that glance in the Moon did glitter.
The man hath penance done, And scarcely he could stand.
How they seemed to fill the sea and the sea.
He hath a cushion plump: It is the ribbed sea-sand.
Each corse lay flat, And the shadow of the sea.
Till over the mast, Had never passed away.
The look with which they died, Had never passed away.

Figure 1. The ancient mariner Sonnet.

These duplicates were most likely due to a relatively small dataset. For that reason, we experimented with a larger dataset that contained sonnets by Shakespeare. But even with more than 700 lines of poems, the Markov chain still could not produce unique lines. So, we expanded our dataset with another 2300 lines by Shakespeare. The output was already much better but the program didn't seem to generate any novel output. To be considered a sonnet, a poem must contain 10 or 11 syllables per line. The program was now able to generate poems that matched the specifications of a sonnet but it wasn't necessarily creative.

That may express my love may still shine bright.10
Now, while the world, unbless some mother.10
How to divide the conquest of a former sight.10
Love, a liquid prisoner pent in thee lie!10
Those that can see, but no such matter.10
To be death's conquest and make my love away.10
To put fair truth upon so foul a lie.10
The wrinkles which thy glass and tell the day?10
Blessed are you, and let that copy die.10
Take all my poverty: and therefore to be.10
If thou survive my well-contented day, to die!10
O! how I faint when I praise thee?10
Cupid got new fire; my friend and me!10
That god forbid, that thou mayst in me.10

Figure 2. Shakespeare Sonnet with all 10 syllables.

Step 1

In our initial approach, we limited the word count, so that the generated sentences all had exactly 10 syllables. These constraints seemed to hinder the program in generating lots of different results, most sentences were highly similar.

We came up with the idea to use a *brute force* approach. That came down to generating lots of sentences and then making sure the endings rhymed. This approach maintained the original structure that was present in our Shakespeare dataset.

Step 2
After examining examples of poetry, we found that poems are actually split into lines instead of sentences. In our new approach, we used the rhyming ends of lines to create poems. We split up each sentence into lines by colon, semicolon or comma.

Step 3
A system that used sonnets to generate sonnets seemed a bit boring. We came up with the idea to create limericks from Shakespeare's sonnets. Limericks are short poems that make the reader laugh. Limericks use the AABBA rhyme scheme. The A-lines have more syllables than the B-lines.

```
Made more or less by thy side. 8
And keep my drooping eyelids open wide, 8
Will play the mother's part, 6
When you have some part, 6
That sometimes anger thrusts into his hide, 8
```

Figure 3. First Limerick logs. Note that there were still duplicates in this case. Purple is the number of syllables.

As with our earlier approach, the program first generates a bunch of lines. Then it categorizes these into groups of 5, 6, 7, 8 and 9 syllables. Afterwards, it bundles lines that rhyme with each other. Lastly, the program removes lines where the endings are the same word. After going through all these steps the result was printed on the screen.

Step 4
We wanted to give an impression that the poems were written by a real author. Therefore, we used a text file with English renaissance male names. Vintage-style paper was used to give a historical feeling to the reader.

A poem title was still missing at this point. An external synonym dictionary was used to generate a title based on the last word in the poem. The title and ending then formed a circle.

Conclusion
The system generates poems that match the characteristics of a limerick. The 1st, 2nd and 5th lines have between 5 and 7 syllables per line, and the 3rd and 5th lines have between 7 and 9 syllables per line.

We believe that the humorous rhyme of a Limerick creates a nice contrast to the literary masterpieces of Shakespeare. The short format also reduces the difficulty in understanding Shakespeare's ambiguous sentences.

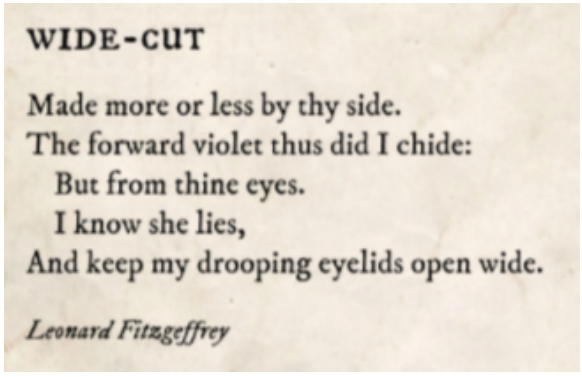

Figure 4. Final Limerick with added style.