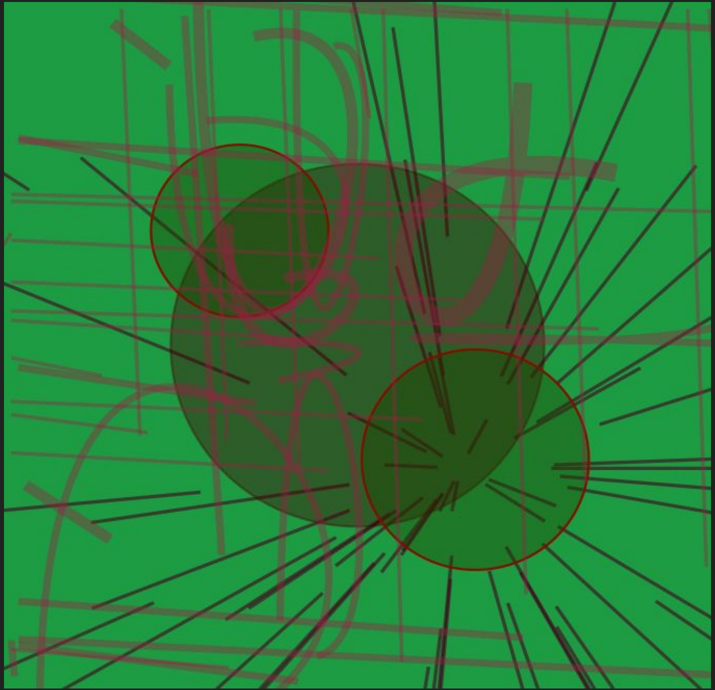
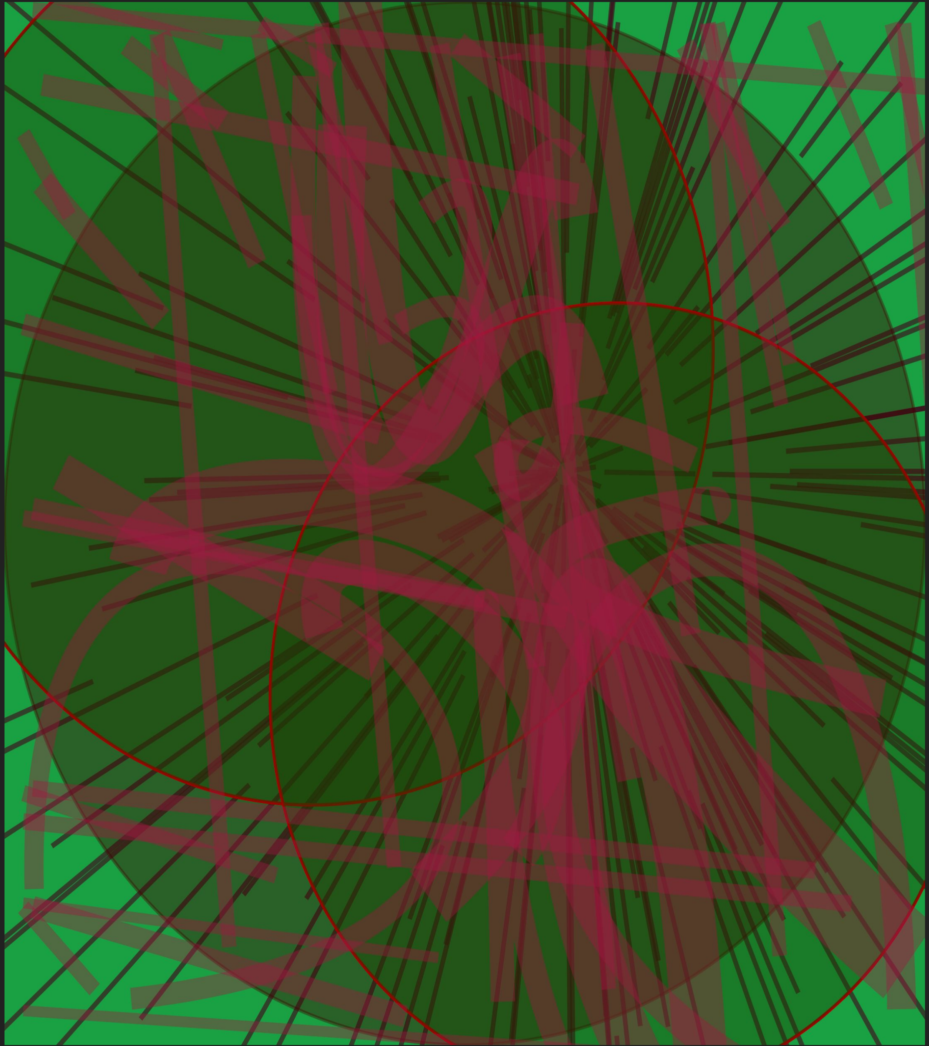
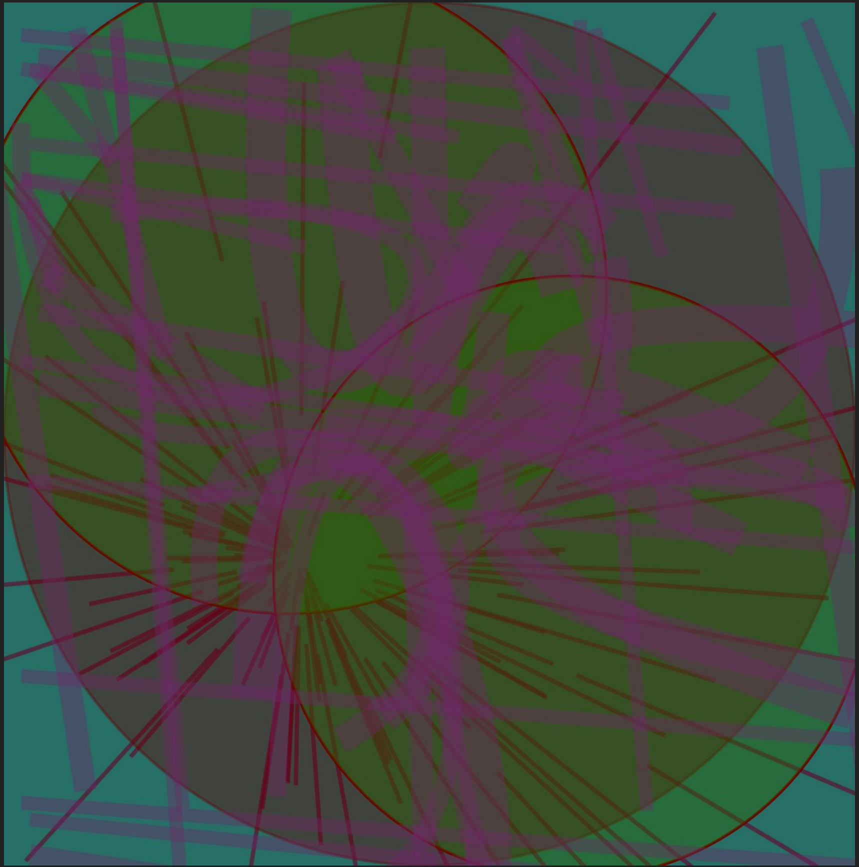
Visualizing Text Conversations

Oberlin Winter Term Project 2019

Final Images







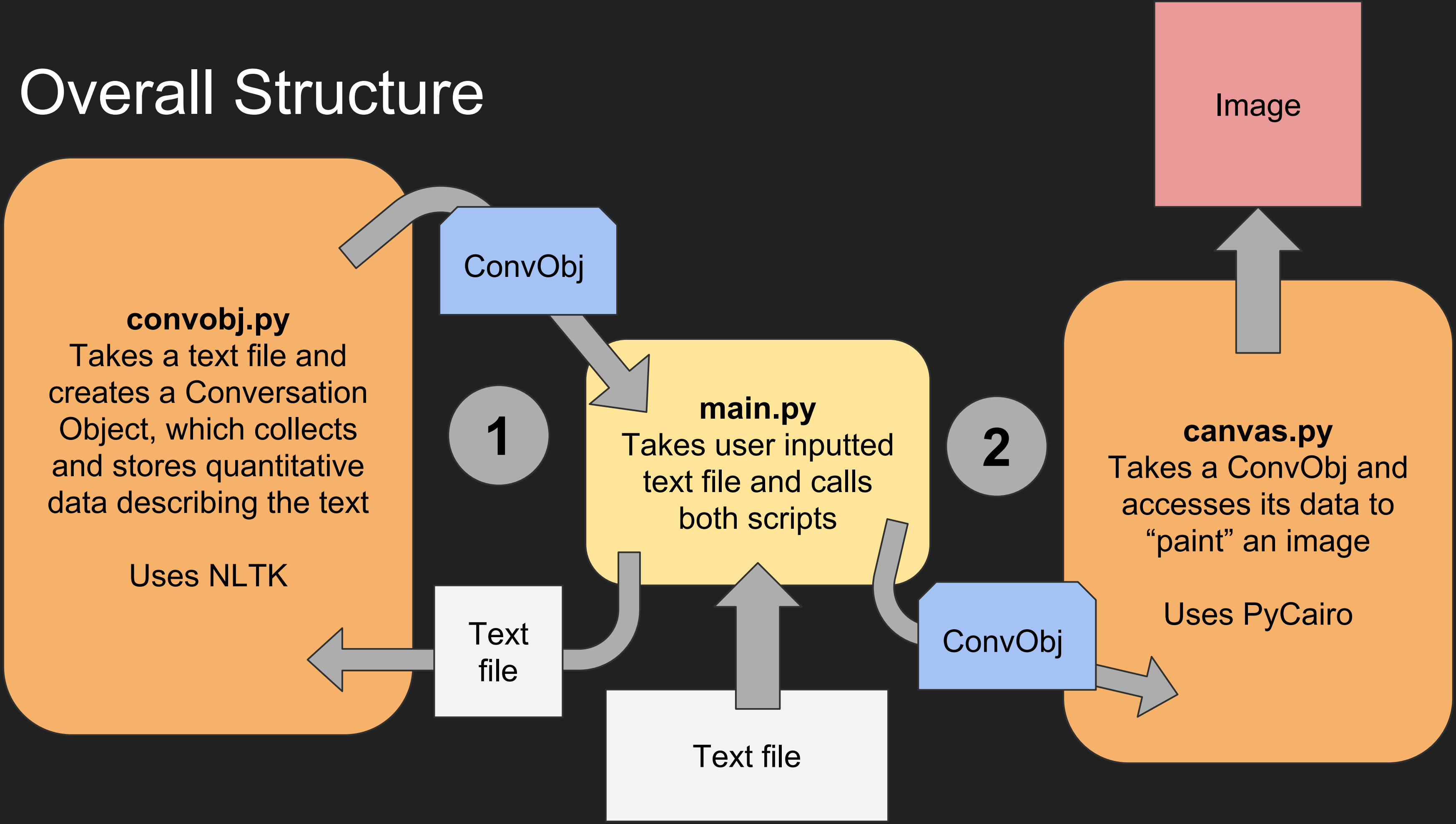
……..but the process is more interesting than the product

Program Anatomy

Technology

- Python
- NLTK (Natural Language Toolkit) for NLP measures
- PyCairo for graphics

Overall Structure



convobj.py
Takes a text file and creates a Conversation Object, which collects and stores quantitative data describing the text

Uses NLTK

ConvObj

Text file

1

main.py
Takes user inputted text file and calls both scripts

Text file

2

ConvObj

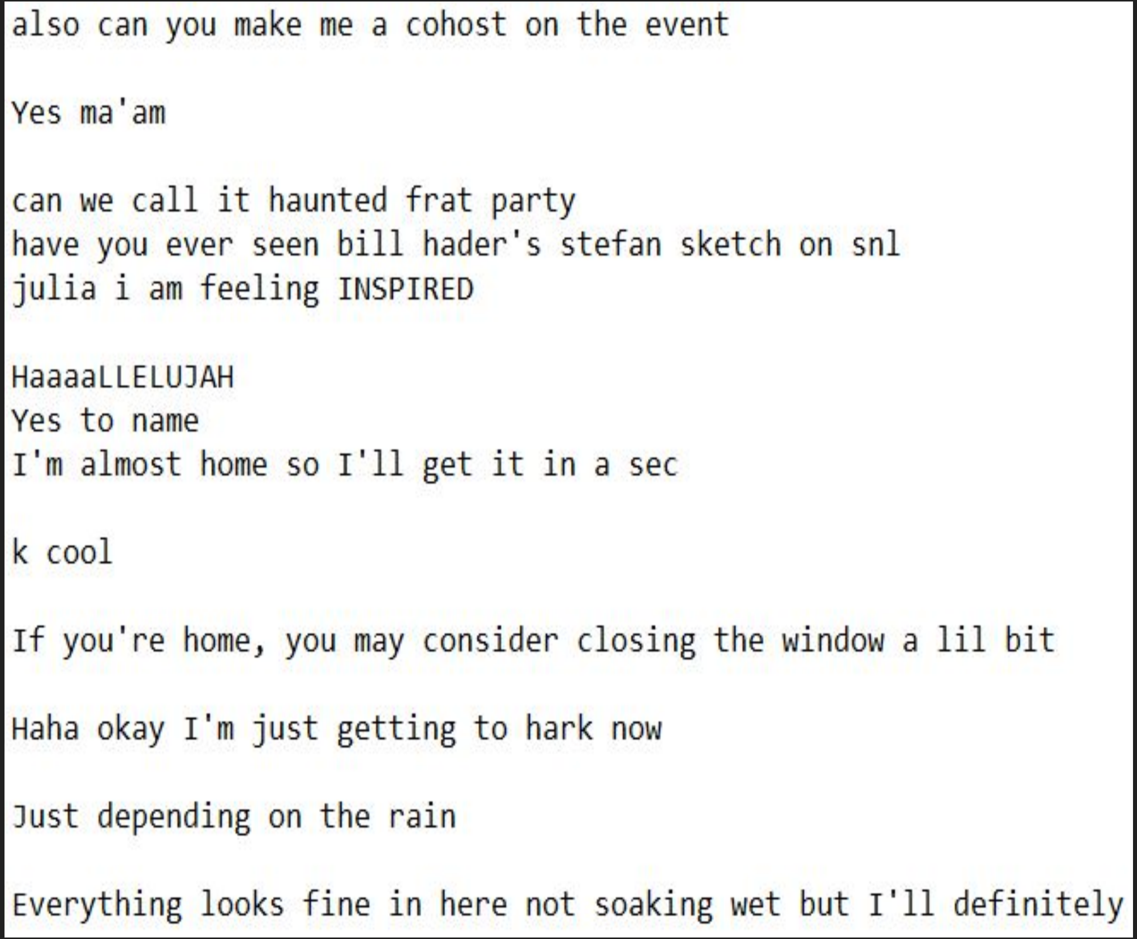
canvas.py
Takes a ConvObj and accesses its data to “paint” an image

Uses PyCairo

Image

Text Files

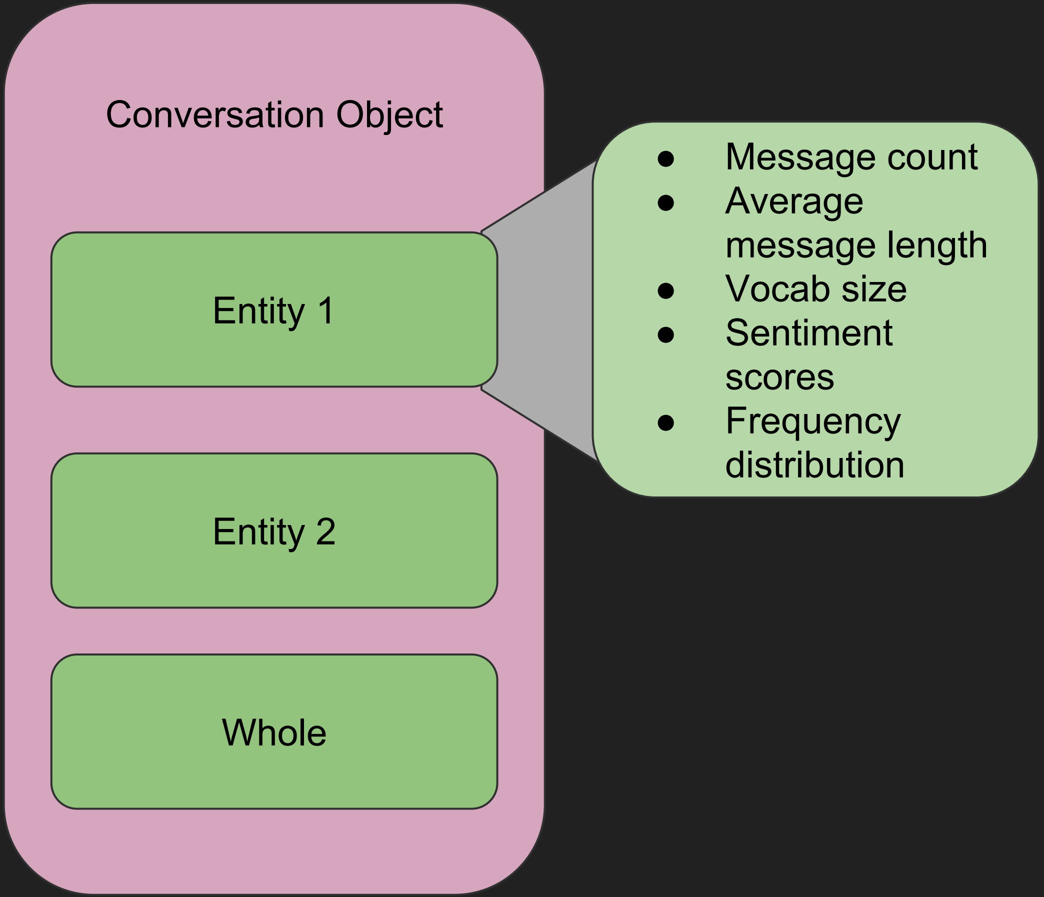
- Mock conversations based on transcriptions of real text conversations (with permission from other speaker)
- In future, have more pre-processing options



```
also can you make me a cohost on the event

Yes ma'am

can we call it haunted frat party
have you ever seen bill hader's stefan sketch on snl
julia i am feeling INSPIRED

HaaaaLLELUJAH
Yes to name
I'm almost home so I'll get it in a sec

k cool

If you're home, you may consider closing the window a lil bit

Haha okay I'm just getting to hark now

Just depending on the rain

Everything looks fine in here not soaking wet but I'll definitely
```

Conversation Object

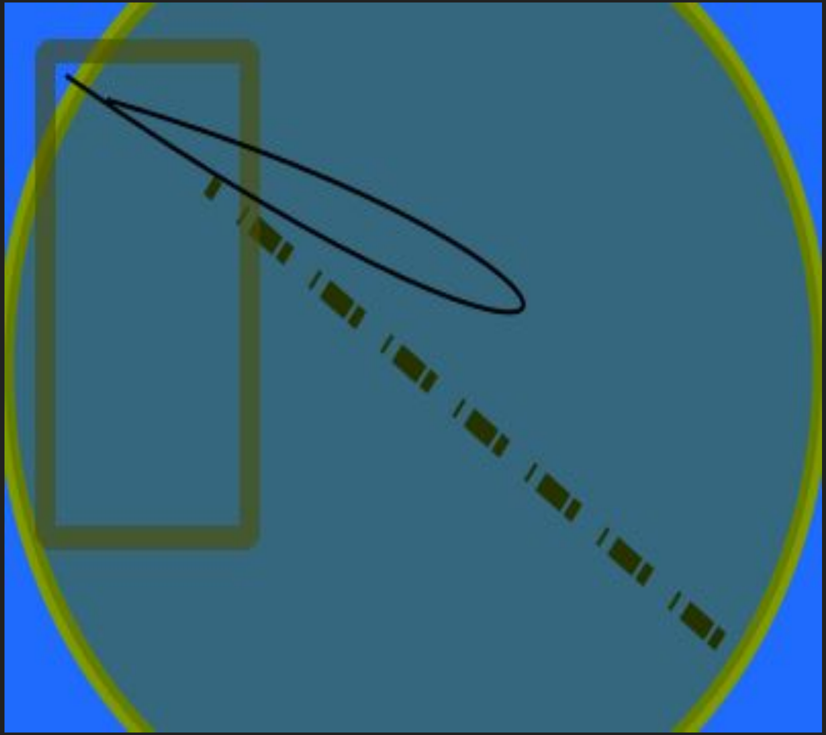
- Isolates the two speakers in strings and creates 3 subclass “Entity” objects: 1 for each speaker and 1 for whole conversation
- Each entity object retrieves quantitative textual data



Artistic Process

Artistic Intent

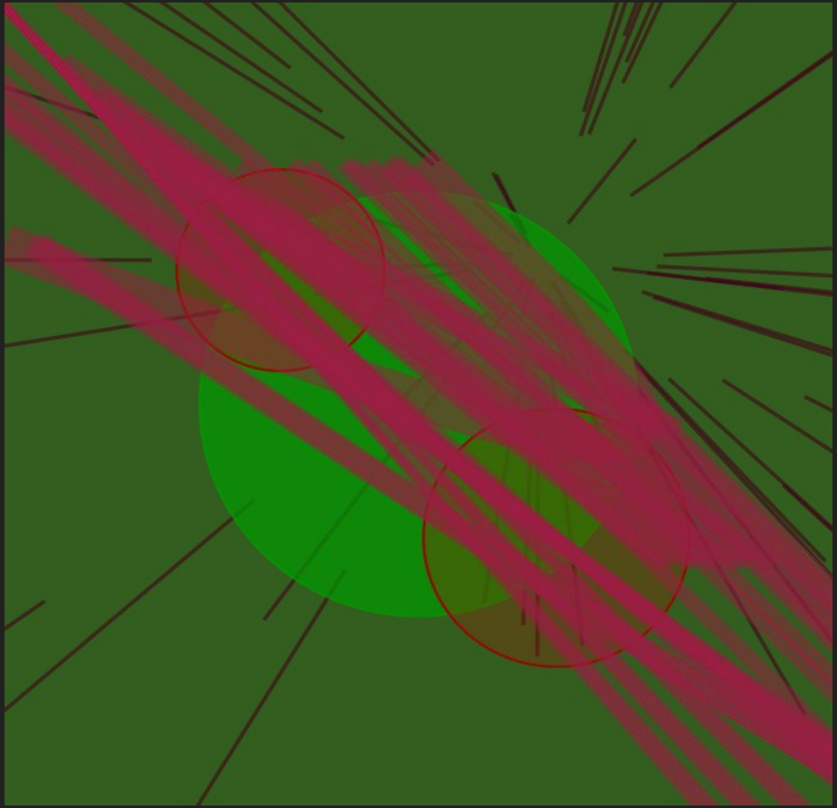
- Once I had the numbers, I had complete creative liberty over how to visualize them within the limits of PyCairo
- Those choices are what make the product art
- Each attribute in entity class maps to unique part of the image
- Wanted to include some randomness so the program output would vary on each run



PyCairo functions test

Sentiment

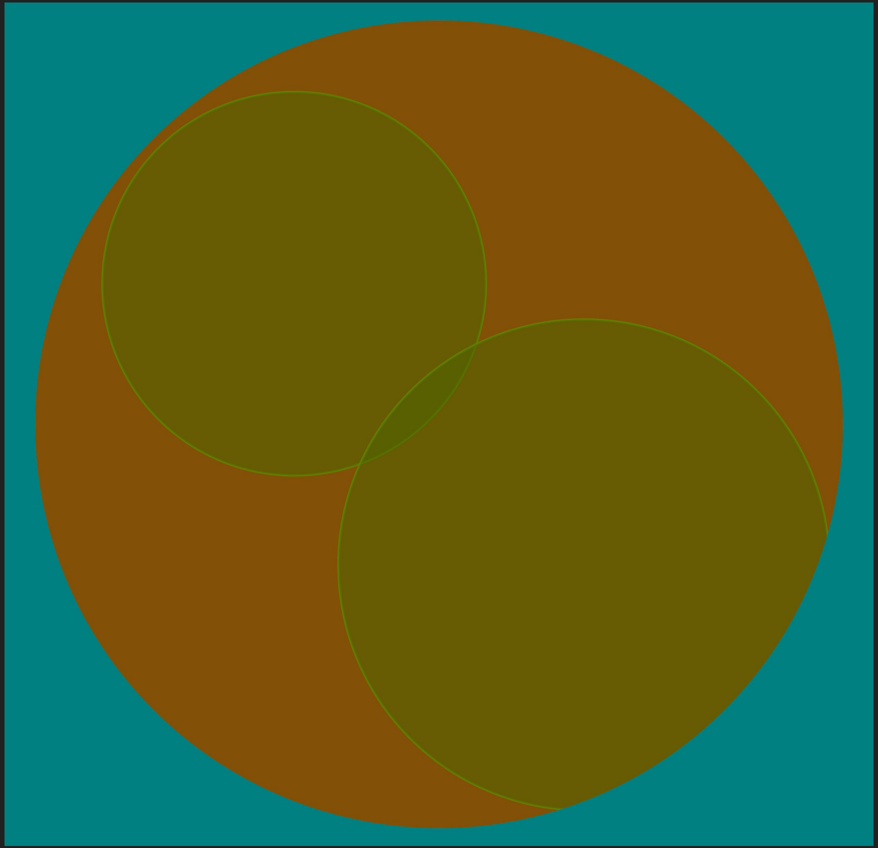
- Sentiment scores determine color palette by mapping positive, neutral, and negative frequency values to R,G,B values
- RGB values created solely from combinations of pos, neu, neg, and 0



All colors generated from sentiment scores

Vocabulary

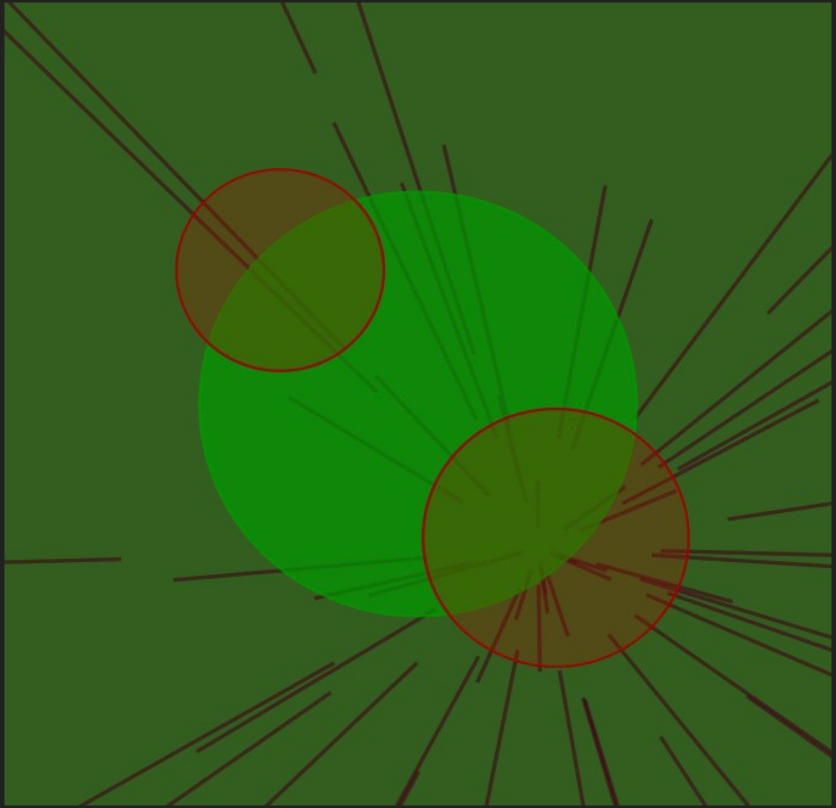
- Circle radii determined by vocab sizes of both entities (inner circles) and combined (larger)
- Centers are hardcoded
- In future, add noise to center positions or make them dependent on other data



Early version

Message Length

- Messages represented as radial lines
- Center offset from center by which entity writes longer messages
- Length and variation of lines determined by average length and standard deviation

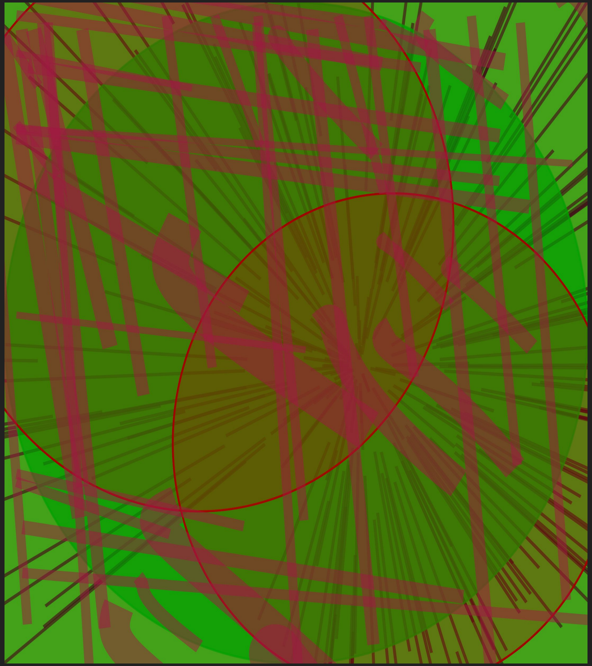


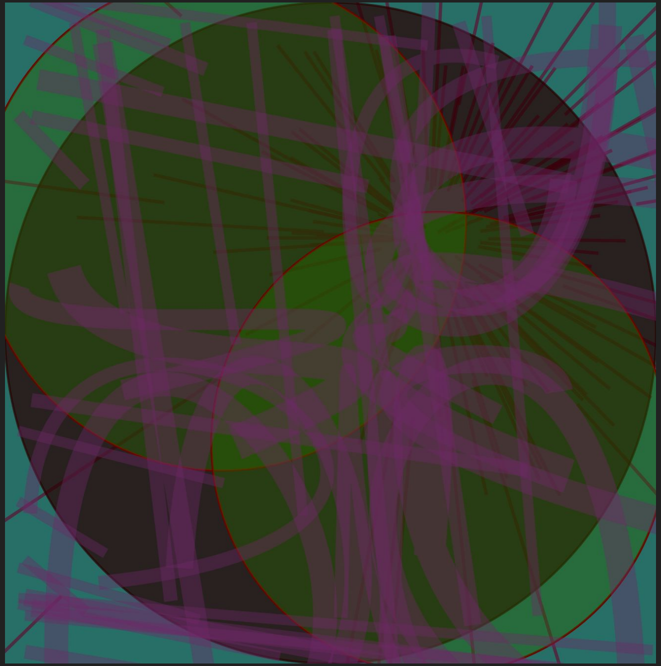
Early version

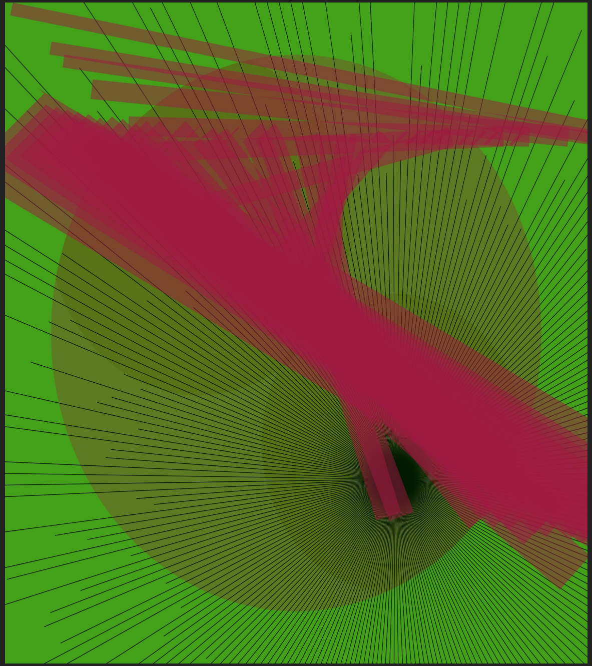
Word Frequency Distribution

- Each word represented in own line or curve
- Start coordinate of line random, end coordinate offset by the freq size in whole conversation, any middle coordinates offset by the freq size for one entity
- Line width depends on freq size
- Feels the most arbitrary. In the future I’d like to create more intentional mappings
- Examples on the next slide...

Word Frequency Distribution







Going Forward

- Explore procedural generation
- Add more measures to the Entity class (e.g. vocab depth, interrogative-ness, expletive-ness)
- Overlay images from text conversation
- Explore GIFs/outputs that would change over the course of reading the conversation

https://github.com/jfriend15/visualize-convo