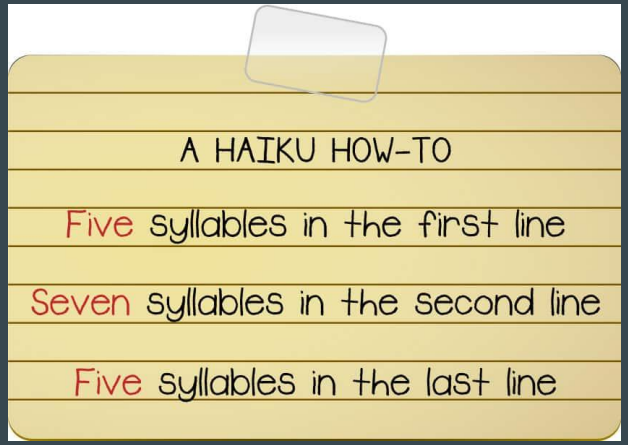
CS 4120 Final Project:

Generating and visualizing Haikus with NLP!

•••

Ben Tunney, Glen Damian Lim

Intro



What is a Haiku?

- A type of short form poetry from Japan
- Composed of 3 lines
- Enforcing the rule of having **17** syllables with a 5-7-5 pattern in each line

Some example Haikus from ChatGPT:

Topic: NLP

Text transformed by code,

Language and meaning entwined,

NLP brings life.

Topic: Pizza

Ate a whole pizza,

Now I am one with the couch,

No regrets, just cheese.

What's our goal here?

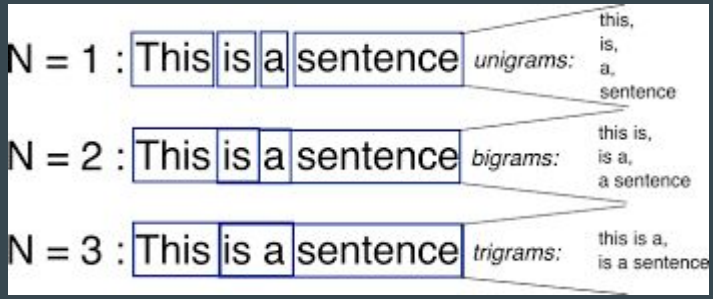
Using NLP in generating Haikus as it is a difficult task (people need to be creative while enforcing the syllables limit in each line). Trying to generate AI art from the produced Haikus.

How?

- Obtaining a Haikus dataset from Kaggle (https://www.kaggle.com/datasets/hjhalani30/haiku-dataset)
- Training a Word2Vec model using vocabularies from the dataset for our word embeddings
- Experimenting with Statistical Language Models and Neural Networks
- Experimenting with external APIs and tools to generate AI
- Creating a dashboard UI for user to generate new Haikus and visualizing them

Statistical Language Model

- Creating a Haiku generator by re-using our n-gram language model from HW 2
- Finding similar words to input and get haikus that include those words as our training set
- Enforcing syllables limit using “syllables” library from PyPI
- Generated Haikus on basketball are enforcing 7 syllables but sentences on line seem to be cut off
- Issues in the dataset (not enforcing syllables limit in each line)




```
i can not wait for
nba young boy hard but he not
same a every


honestly way more
taking the day off and just
nfl i love this time


crazy thinking bout
this is why houston is the
i could listen to


hockey family
tell me two baseball player
ca decide if


playoff hockey is still
have watched the playoff this year
join u on the nba
```

Neural Networks

Recurrent Neural Networks (RNNs): long short-term memory (LSTMs) networks

- RNNs could store long term information but potentially suffer from the vanishing gradient problem
- LSTMs solve the problem above as it the extra layer could selectively choose information
- Produces more meaningful Haikus compared to our n-gram language model

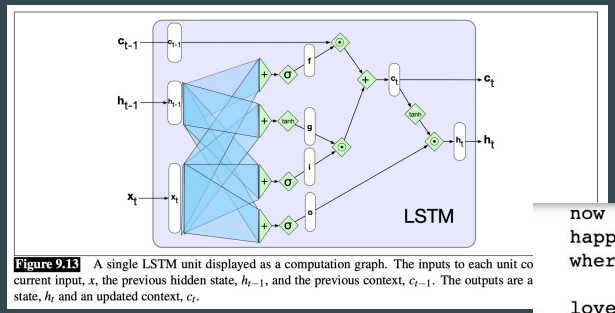


Figure 9.13 A single LSTM unit displayed as a computation graph. The inputs to each unit co current input, x, the previous hidden state, h_{t-1}, and the previous context, c_{t-1}. The outputs are a state, h_t and an updated context, c_t.

```
now when they call you
happy belated birthday
where god is blessed

love ya next week also
i so damn happy with u
cabello love you

i only way to
blessing blessed evening
man i literally

belated express
what should be amazing man
it wa blessed a

if you and loved
good one with you might get
the same song so you
```

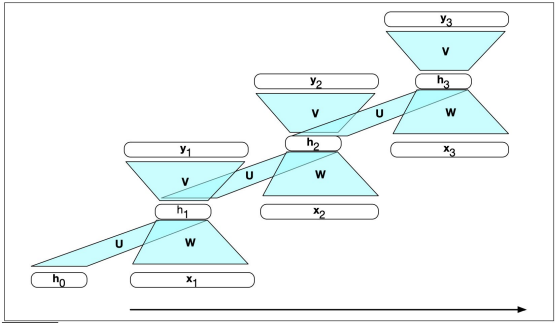
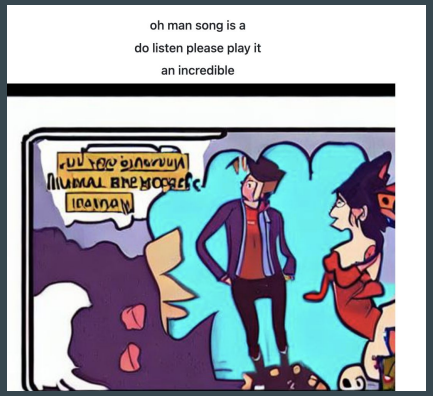


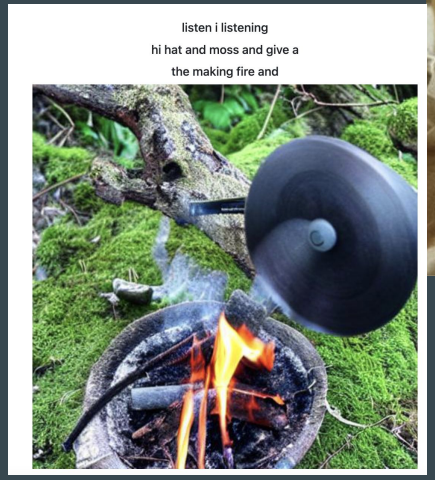
Figure 9.4 A simple recurrent neural network shown unrolled in time. Network layers are recalculated for each time step, while the weights **U**, **V** and **W** are shared across all time steps.

Generating AI Art

Hugging Face implementation of Stable Diffusion in Runway ML

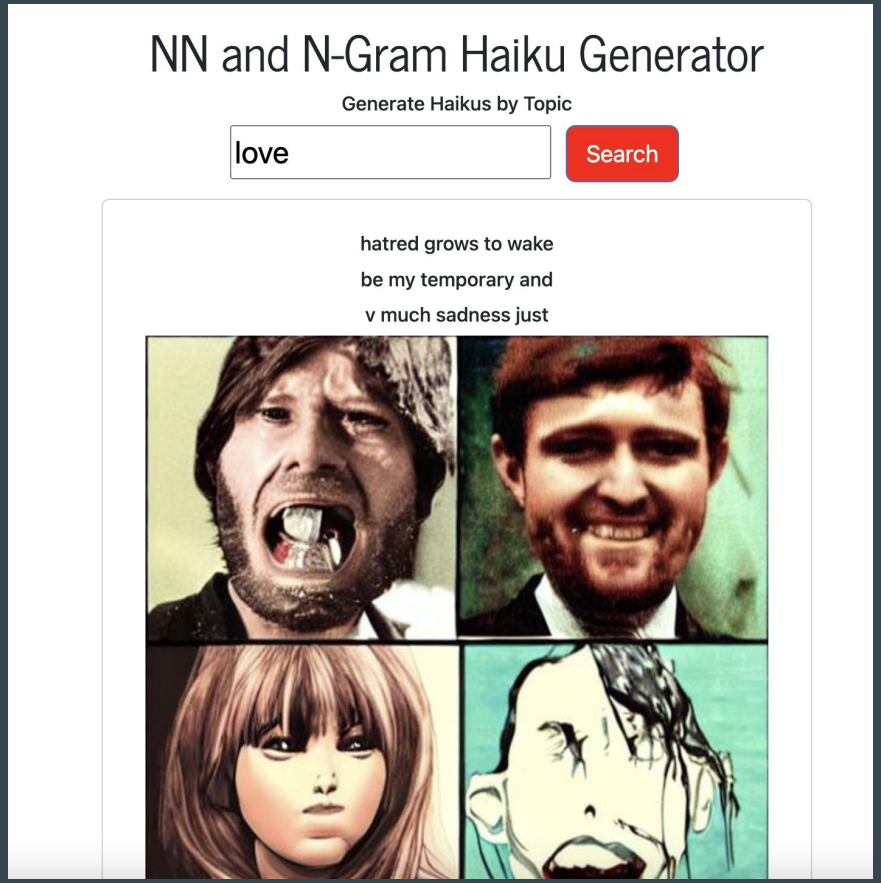
- Stable diffusion maintains performance of ML models
- Uses a latent diffusion model
- Trained on 512x512 images, uses text encoder
- Implemented in Python, haiku prompts
  - NER, Stopwords





Plotly Dashboard

- Query by topic for both our LM and RNN models
- Haiku-based image generation supported by an external API from Runway ML

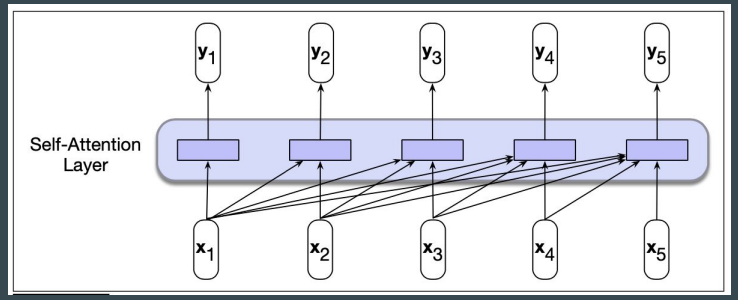


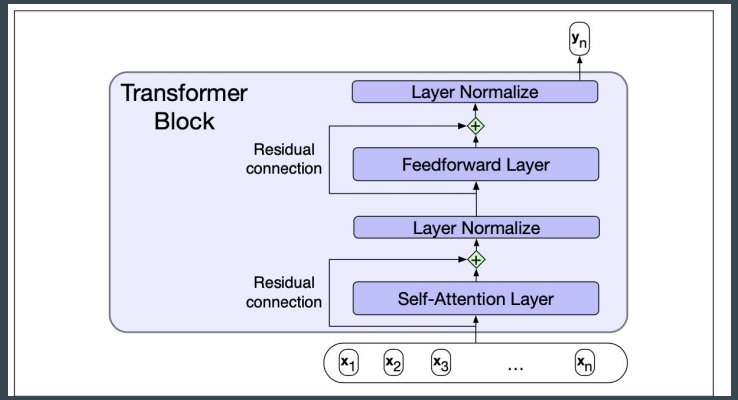
Conclusion

- Our model is hugely affected by the dataset, would produce better result with a more accurate data
  - Specific tasks with constraints such as syllables require large, specific datasets
- Training RNNs are challenging, especially the part when we are pre-processing our data to feed it to the network

Future Improvements

- Using a better model, with more specialized transformer
- Improved UI with new mood visualization
- Multiple words/sentences input queries
- Training our own generative AI model





Thank You!