EDGAR ALLAN POE-TRY GENERATOR

David Heffren, Matthew Sutton, Liliana Mack

INTRODUCTION

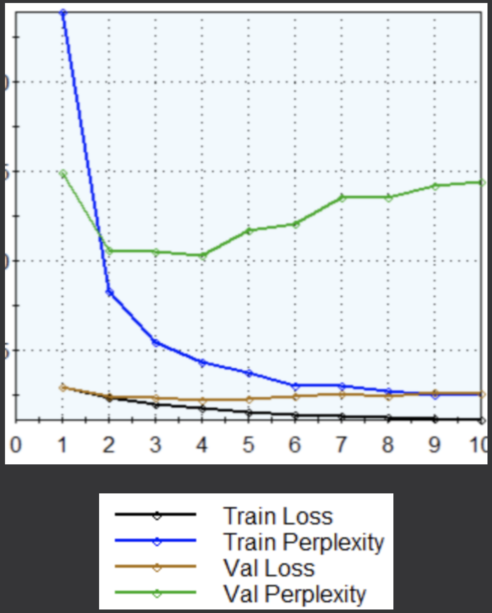
The goal of this project was to generate poetry in the style of Edgar Allan Poe using the pretrained language model GPT2, and fine tuning it to our specific task.

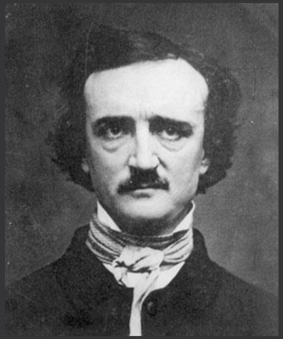
This project was based on the paper GPoet-2, which did the same task but to generate limericks.

The target goal was just to emulate the theme/style of his poems, whereas the stretch goal was to actually generate structurally coherent poems.

RESULTS

Our results outperformed a random model, but they failed to generate coherent poetry. You could tell that it was taking influence from Poe if given a Poe-like prompt. We ended up with a validation cross-entropy loss of 2.5872 and a validation perplexity of 14.4569.





DISCUSSION

The limitation of our model was our inability to get coherent poems, even though we were able to somewhat replicate his style.

For further work in this area, we would try to replicate the structure of Poe's poems, or just poems in general, by using the Reverse LM method described in the paper. We would also try to use more computational resources

We learned that often when there is an existing architecture for a problem, sometimes the best solution is scaling it to fit your needs.

GENERATED POEMS:
I hated to hear the sound of my beating heart (PROMPT)
and, the bells that on need door at nightly and glance
burning poets at any make, to here by onal w dids more in the terminating'

Once upon a midnight dreary
oth len from and to me to the danger down
but the nebabel my angels lot of lake did at fair and a heaven at kind
 a amid ch he my throne and with you well
a autumn no bes stars, light, half cried the night

METHODOLOGY:

Dataset:
Our Dataset for this project was the collection of all of Edgar Allan Poe's poems(49), each loaded from a text file, and formatted for clarity.

However, when inputting the data into the model, we had to cut up some poems, which were longer than the max sequence length of 1024.

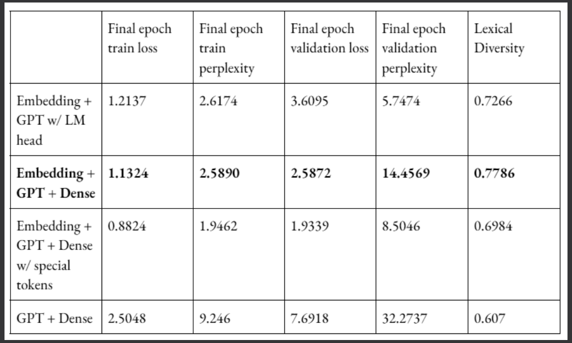
Architecture:
At the core of our project is the GPT2 Architecture, with 117 million pretrained parameters, and is similar to a Decoder based Transformer architecture.

The model has a vocabulary size of 50257 and a max sequence length of 1024.

We froze the pretrained layers and added our own layers before and after to fine tune to our task.

Specifically, although we tried multiple architectures, we went with an embedding layer (replacing that of the GPT2) before the model, and a dense softmax layer after (instead of the default GPT2 classification layer.

The idea of this was to learn our own embeddings which would relate more specifically to Poe's poetry, and to lean towards specific "poe-like" words with the tail.



	Final epoch train loss	Final epoch train perplexity	Final epoch validation loss	Final epoch validation perplexity	Lexical Diversity
Embedding + GPT w/ LM head	1.2137	2.6174	3.6095	5.7474	0.7266
Embedding + GPT + Dense	**1.1324**	**2.5890**	**2.5872**	**14.4569**	**0.7786**
Embedding + GPT + Dense w/ special tokens	0.8824	1.9462	1.9339	8.5046	0.6984
GPT + Dense	2.5048	9.246	7.6918	32.2737	0.607