6-17-24-CL Pipeline
Objective: Develop full training system

Analysis original based on bigchat: 6-10-24-BigChat Analysis

- Core data: Using critical role dataset, git
- Support data: Dolly15k Dataset, arena dataset, open orca, orca math
- Extra possible data: gov hearings
- Using Dolly V2 and pythia models allowing work with pretrained types or instruction tuned

Plan:
1. Explore CR3 Dataset
2. Setup Dolly V2 and pythia for inference then training
3. Setup train inference pipeline (use analysis work)

- Set up summaries as QA pair scripts.

- training accuracies to follow:
    - General data
    - Questions on chunks
    - Response accuracy
    - Total inference output similarity size

6-24-24
Previous work looked over and got the new CR3 dataset to be working on the new embedding and analysis methods. They seem promising enough and it's time to try to connect something things together.

I want to get the full pipeline up and running. I'll start with some very small tests, very small model and not worry about results as of now just make a simple system to test the structuring.

CR3 Dataset: Long running dataset of D&D explored and run in previous document
Inference: Distributed high throughput inference will test but maybe ~1K inference per text
OpenAI Embed: Use openai embedding models to embed the responses to analyze
AE Anlaysis: Learn what the best data is in the responses and select them
Dataset Create: Make a new dataset for training that is a ratio of different datatypes
Train: train
Test: Questions around the previous step, general performance, long term memory
Repeat above steps for number of pipeline steps.

CR3 Dataset	Inference	OpenAI Embed	AE Analysis	Dataset Create	Train	Test

- Global
    - Experiment ID
    - Configuration
    - Current progress

- CR3 data
    - Utterance Block to inference
    - Utterance response to target
    - File/order
    - Step

- Inference/Embedding Data
    - Input, run data (time to finish, inferences, ect)
    - Responses, configurations (temp,top_p,len)
    - Embeddings (combine steps)
    - Connection to CR3 Data

- AE Analysis
    - AE models
    - Umap, kmeans, dbscan, pca processed raw data
    - Visualizations
    - Metrics/stats
    - Selected responses
    - Connection to inference/embedding

- Dataset Creation
    - Configuration per dataset (ratios, selections)
    - Dataset: Combination of multiple datasets in varying ratios
    - Connection to AE data

- Train
    - Model at step
    - Training metrics
    - Time, gpu utilization

- Test
    - Configuration
    - Does this repeat Inference/embed and AE analysis?
    - Accuracy to correct response

GLOBAL NOTES:

Dataset contains dice rolls, the results of these dice rolls should not be penalized on how the model responds.

Dataset is messy in that it has repeated utterances from the same person. Back to back. Easy fix may be to combine this data.

Progress

- Global
  N/A

- CR3 data
  Initial analysis of data, large embedding analysis complete, AE functional

- Inference/Embedding Data
  Submodules work need to combine and performance test

- AE Analysis
  Questions around time and usability. Need to compare to standard methods.

- Dataset Creation
  Dolly15k functional, need more data and storage of previous step recall.

- Train
  Training well, dolly15k data is poorly training, could be test model size

- Test
  N/A

6-25-24
Rough idea as of now:

Structure:

```
my_project/
├── config/
│   └── config.yaml
├── cr3_data_module/
│   ├── __init__.py
│   ├── data_setup.py
├── inference_module/
│   ├── __init__.py
│   ├── inference.py
│   ├── embedding.py
├── ae_analysis_module/
│   ├── __init__.py
│   ├── autoencoder.py
│   ├── analysis.py
├── dataset_creation_module/
│   ├── __init__.py
│   ├── create_dataset.py
├── training_module/
│   ├── __init__.py
│   ├── training.py
├── testing_module/
│   ├── __init__.py
│   ├── testing.py
├── scripts/
│   └── run_pipeline.py
├── utils/
│   ├── __init__.py
│   ├── logging.py
│   ├── config.py
└── README.md
```

Pipeline will allow stopping and starting at will to allow debugging and design at all stages.

First working on integrating inference and embedding into the pipeline

7-1-24: Pipeline

Plan

1. Put individual scripts run separately and unconnected into the pipeline and have it run. Ensure everything is saving in the correct location.
2. Step by step start connecting the individual steps together to create the final system.

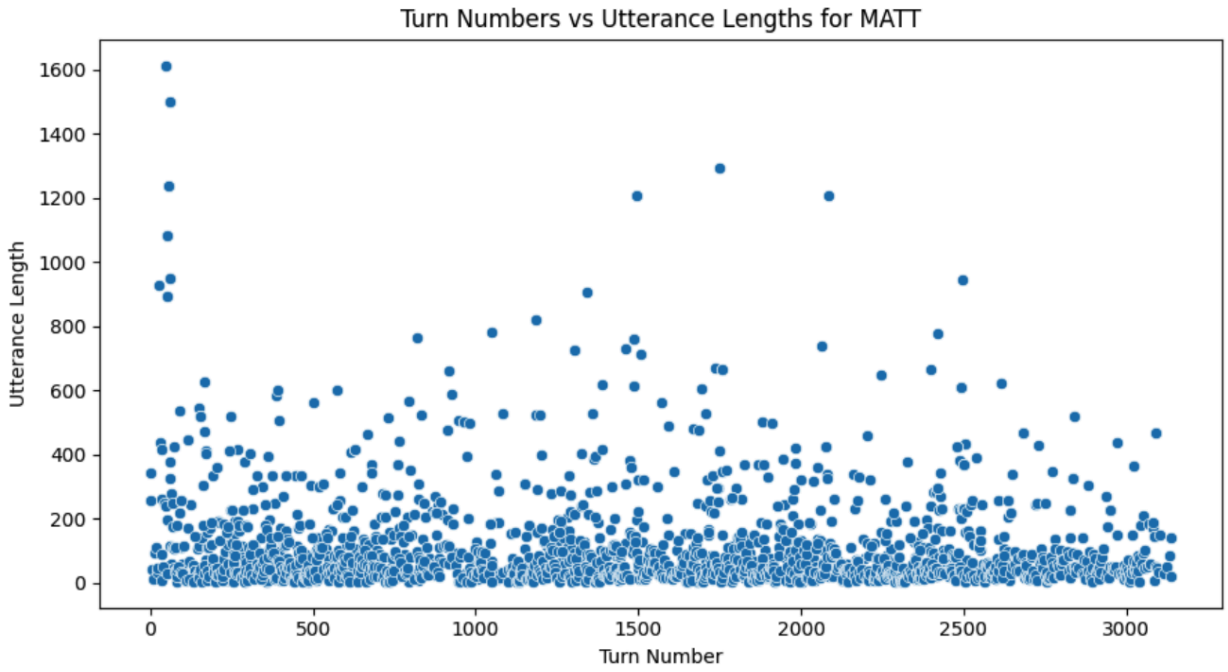
7-2-24
- all individual steps running in pipeline no error
- Independent and not connected at all

Problems:
- data is messy
- there are nearly 3 million turns in the data, for someone like MATT there is 800k turns

This makes it nearly impossible to use all the data as the per step basis is not fast enough.

Single files have around 2k turns inside of them

How do I deal with massively variable prompt sizes?



The variability in number of steps to the character's utterance is sometimes very high, other times it is back to back (that needs to be combined or removed)
Min Turn Difference: 0
Max Turn Difference: 41

This means sometimes the prompt will be small, other times it will be massive

TODO DATA:
- select a few files
- clean data (remove extra info other than game)
- build test data and questions

Plan:
- use text mask for everything but the character of choices response
- combine same character back to back responses
- clean data (ignore all summary data it sucks)
- new simpler data format

Per step:

1. Train with masks of all text leading up to our character's first statement
    a. Balance the number of masked tasks with general open source chatbot tasks
2. Inference on the chatbot style response for character in large fashion
3. Train on filtered down chatbot inferences
    a. Balance with random general data that is masked (the pile?)

I should have a minimum of 5 previous steps in the pretraining step even if it has the character's name in it. Otherwise it will be training on a single entry of like 2 words.

I will also need to combine or manipulate responses that are very small. I don't want to work with data that is only around 1-3 words

STOP TO PERFORM RARE MEMORY SUBMODULE STUDY

Solid validation implies rare memory injection works as hypothesized. Early in the pipeline I may need to increase number of inferences and select rarer more similar outputs in order to truly tune it. 7-3-24 LR Studies/Predictors

Working on initial data function:

Come to find out there was a ton of repeated data in the folder. Like a lot! I cleaned up the repeated folders and for just C1 there is “only” 242420 turns which is an exponentially less amount than before.

```
Summary Metrics:
Total Files: 94
Total Chunks: 211925
Total Turns: 211925
Total Words: 2957162
Unique Names: CHRIS HARDWICK, OFFSCREEN, DARIN, MATT
INE, CHRIS PERKINS, TALIESIN, ZAC, DENISE, KIT, IVAN
T, ORION, IFY, JOE, TRAVIS, ALEC, NOELLE, ALL – ORIC
Number of Unique Names: 51
Average Chunks per File: 2254.52
Average Turns per File: 2254.52
Average Words per Utterance: 13.95
Common Names: MATT, MARISHA
Average Steps Between Utterances for Common Names:
  MATT: 0.04
  MARISHA: 0.10
```