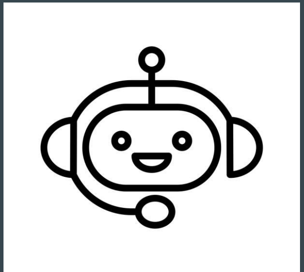
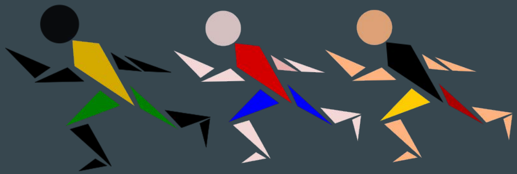


Summer 2024 Olympics Question Answering (QA) Model

Fine-Tuning LLMs for Domain-Specific QA Tasks

Carrie Aponte
Team 14



PROBLEM

Creating a QA Summer 2024 Olympics ChatBot

- Delivering accurate information on recent events

Existing Approaches:

- Training and evaluating multiple LLMs and transformers on QA tasks
- Training LLMs and transformers on domain-specific dataset

DATA / TASK

Data:

- Summer 2024 Olympics Dataset
  - 58 csv files - 13 general, 45 results
  - 200,000+ QA Pairs
- SQuAD
  - 100,000+ QA Pairs
  - Training for general QA task

Task:

- Transform Olympics dataset to appropriate QA format
- Train LLM on dataset
- Respond to queries with information from provided dataset

	code	name	name_sho	name_tv	gender	function	country_cc	country	country_lo	nationality	nationality_	nationality
2	1532872	ALEKSANY	ALEKSANY	Artur ALEK	Male	Athlete	ARM	Armenia	Armenia	Armenia	Armenia	ARM
3	1532873	AMOYAN N	AMOYAN N	Malkhas AI	Male	Athlete	ARM	Armenia	Armenia	Armenia	Armenia	ARM
4	1532874	GALSTYAN	GALSTYAN	Slavik GAL:	Male	Athlete	ARM	Armenia	Armenia	Armenia	Armenia	ARM
5	1532944	HARUTYUN	HARUTYUN	Arsen HAR	Male	Athlete	ARM	Armenia	Armenia	Armenia	Armenia	ARM
6	1532945	TEVANYAN	TEVANYAN	Vazgen TE\	Male	Athlete	ARM	Armenia	Armenia	Armenia	Armenia	ARM

DATA CLEANING / PROCESSING

Investigated data

Selected datasets

Selected columns

- Relevance

Handled missing values

- Replaced with “”

Normalized

Transformed to single string

```
coaches_columns = ['code','name', 'gender', 'function', 'country_code', 'country_long',
coaches_data = coaches_data[coaches_columns]

coaches_data = coaches_data.fillna("")

coaches_data = coaches_data.map(lambda x: x.lower() if isinstance(x, str) else x)

coaches_data['text'] = coaches_data.apply(
    lambda row: ' | '.join([f"{col}: {row[col]}" for col in row.index]), axis=1
)

coaches_data = coaches_data[['text']]

coaches_data.head(150)

coaches_data.to_csv('/content/drive/MyDrive/Courses/CIS531/Term_Project/olympics/Process
```

1	text
2	code: 1533246 | name: pedrero ofelia | gender: female | function: coach | country_long: mexico | disciplines: artistic swimming |
3	code: 1535775 | name: radhi shenaishil | gender: male | function: head coach | country_long: iraq | disciplines: football | events:
4	code: 1536055 | name: aflakikhamseh majid | gender: male | function: coach | country_long: islamic republic of iran | disciplines:

DATA WRANGLING CONT.

Convert to df

Parsed information

- Created dictionary

Cleaned dictionary

- Removed extra characters, normalized

Generated QA entries:

- context, question, answer

Cleaned answers

```
[ ] def parse_info(text):
        pattern = r"(\w+):\s(.*?)\s(?=\w+:|$)"
        matches = re.findall(pattern, text)
        return {key.strip().lower(): value.strip() for key, value in matches}
```

```
[ ] athletes_df['parsed'] = athletes_df['text'].apply(parse_info)

    athletes_df['parsed'].head()
```

```
if "nickname" in cleaned_data and cleaned_data["nickname"]:
    qa_entries.append({
        "context": context,
        "question": f"What is the nickname of {name}?",
        "answer": cleaned_data["nickname"]
    })
if "disciplines" in cleaned_data and cleaned_data["disciplines"]:
    qa_entries.append({
        "context": context,
        "question": f"What are the disciplines of {name}?",
        "answer": cleaned_data["disciplines"]
    })
if "events" in cleaned_data and cleaned_data["events"]:
    qa_entries.append({
        "context": context,
        "question": f"What events does {name} compete in?",
        "answer": cleaned_data["events"]
    })
```

RESULTING QA DATASET

List of df from csv files

Combined with pd.concat()

Output one file as the full QA dataset

```
qa_dataframes = []

for input_folder in input_folders:
    qa_files = [f for f in os.listdir(input_folder) if f.endswith('.csv')]
    for qa_file in qa_files:
        file_path = os.path.join(input_folder, qa_file)
        try:
            df = pd.read_csv(file_path)
            qa_dataframes.append(df)
            print(f"Successfully read {qa_file} from {input_folder}")
        except Exception as e:
            print(f"Failed to read {qa_file} from {input_folder}: {e}")
```

	A	B	C	D	E
1	context	question	answer		
2	date: 2024-07-31 at 20	What was the result for dunkel nils (r	13.7 points		
3	date: 2024-07-31 at 20	What type of result did dunkel nils (m	points		
4	date: 2024-07-31 at 20	In which event did dunkel nils (men's	mens all-around artistic gymnastics		
5	date: 2024-07-31 at 20	Where did dunkel nils (men's all-arou	bercy arena		
6	date: 2024-07-31 at 20	When did dunkel nils (men's all-arou	2024-07-31 at 20:13:54 +0200		
7	date: 2024-07-31 at 20	What was the result for rijken frank (	13.733 points		
8	date: 2024-07-31 at 20	What type of result did rijken frank (n	points		

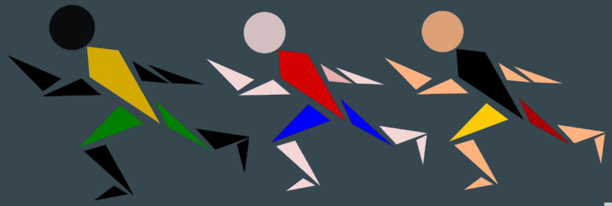
FURTHER DATA MODIFICATION FOR MODEL TRAINING

BERT models required indexing into the context and were not flexible

- Handled by dropping affected rows
- Had enough data still

Tokenizing and preprocessing data for BERT was difficult

- Other models did not require as much data manipulation



APPROACH

Models:

- gpt2 models, flan-t5, BERT models

Techniques:

- Data wrangling, Custom QA dataset creation, Data Merging
- QLoRA quantization, Hugging Face Transformers, Trainer API,

EXPERIMENTAL SETUP

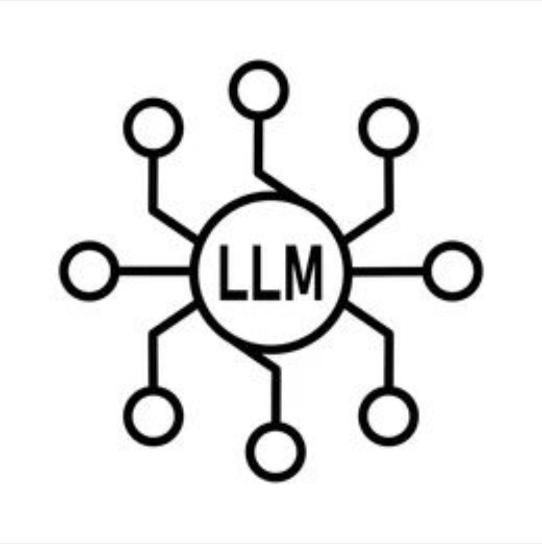
Research Questions:

- Which model performs the best in QA tasks?
- Will fine-tuning with a generated QA Olympics dataset improve QA performance?

Evaluation Metrics:

- BLEU
- ROUGE
- BERT Score
- Training Loss

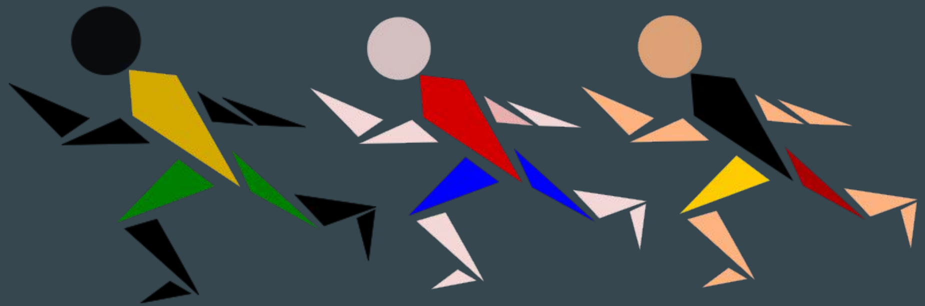
MODELS AND EVALUATION / RESULTS



LANGUAGE MODELS EXPLORED

- gpt2-large
- gpt2-xl
- Flan-t5

- BERT
- RoBERTa
- ALBERT



BASELINE GPT2-LARGE

Evaluated non-fine-tuned gpt2-large on my Olympics dataset:

BLEU Score: 1.4527

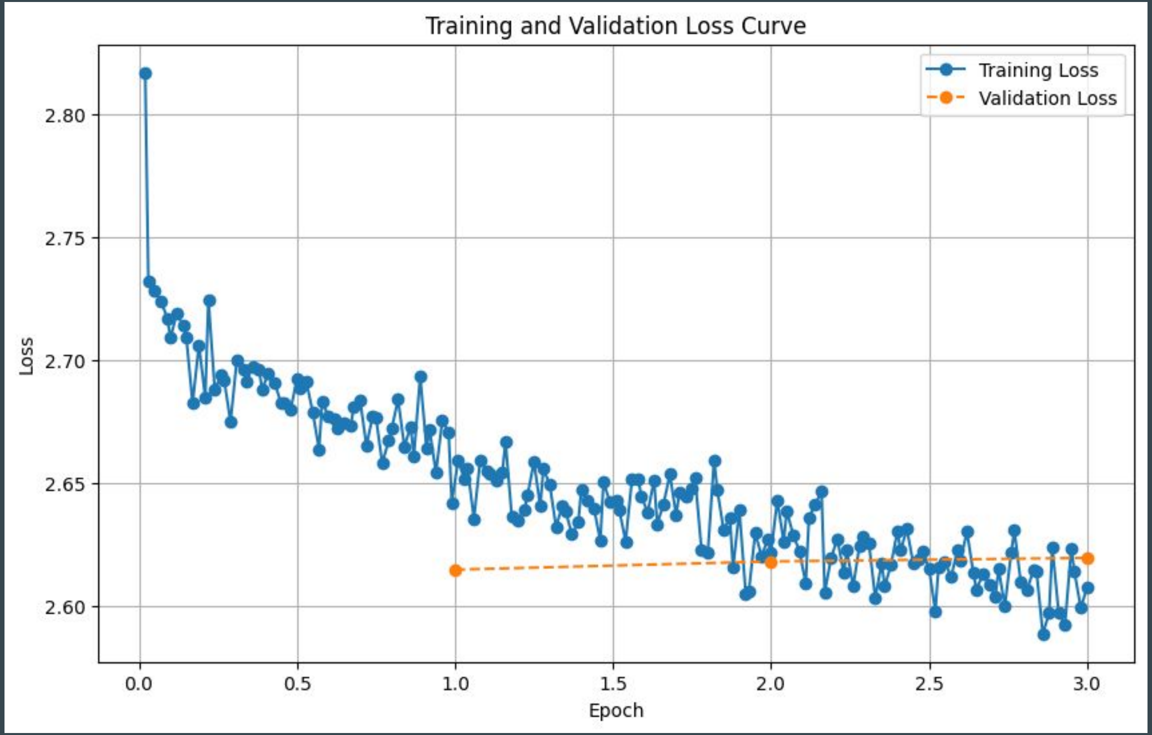
ROUGE1 Score: 0.0505

ROUGE2 Score: 0.0342

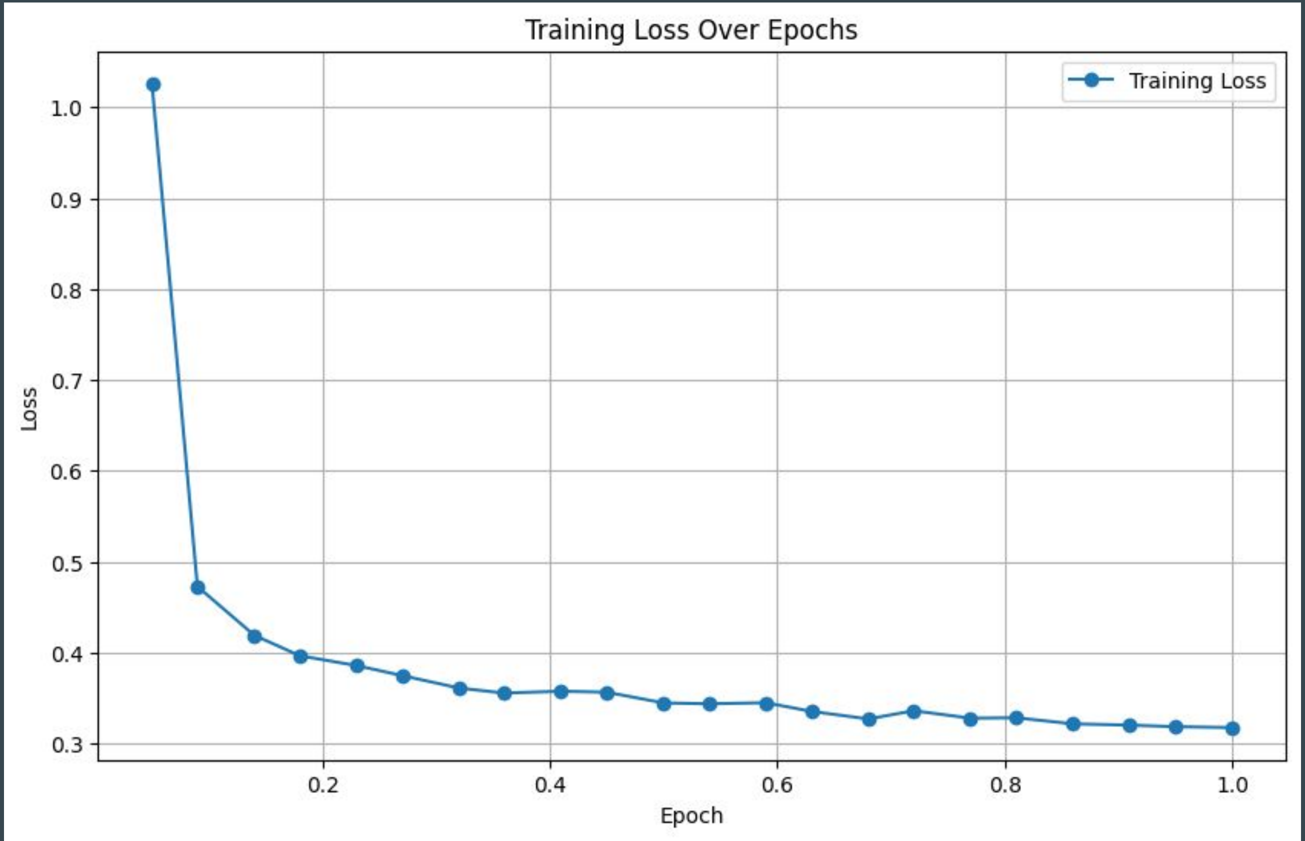
ROUGEL Score: 0.0505

GPT2-LARGE, ONLY SQuAD

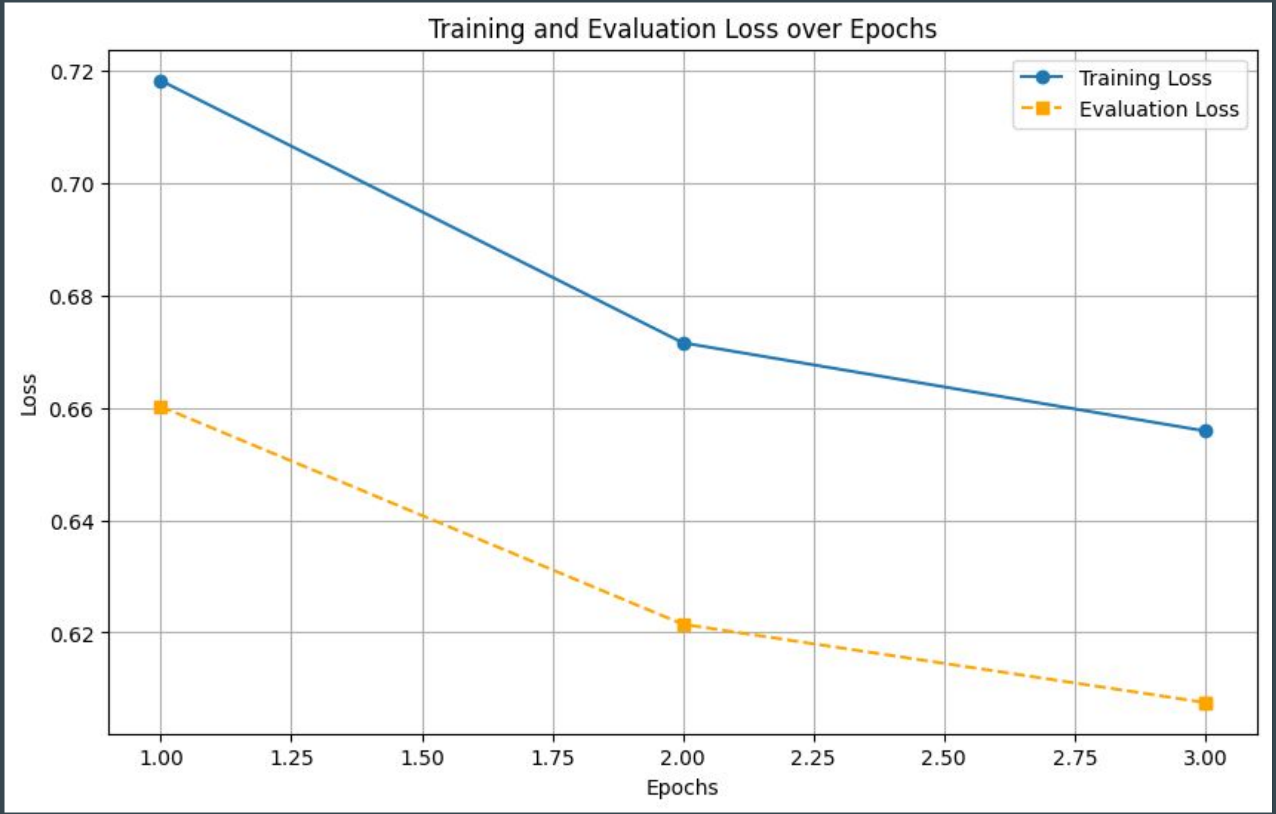
Epoch	Training Loss	Test Loss
1	2.64	2.61
2	2.63	2.62
3	2.61	2.62



GPT2-LARGE WITH SQuAD AND OLYMPICS

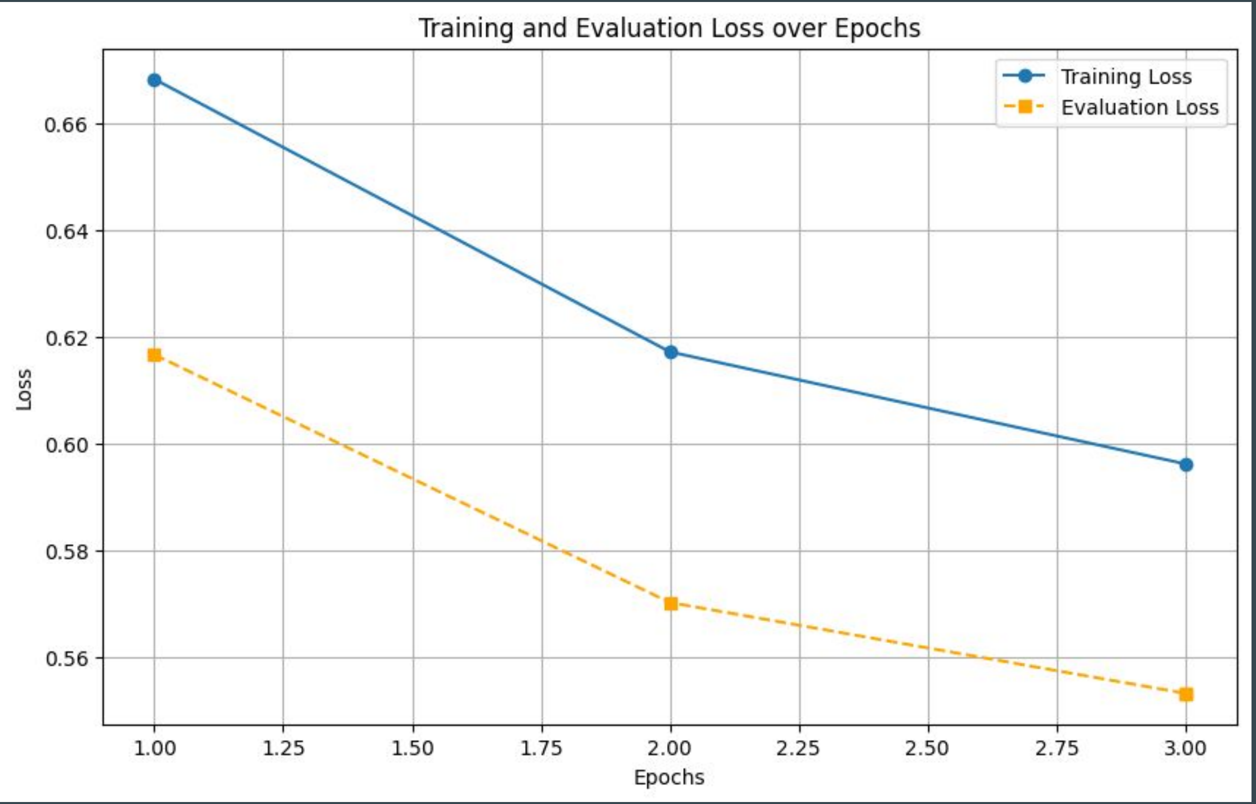


GPT2-LARGE JUST OLYMPICS

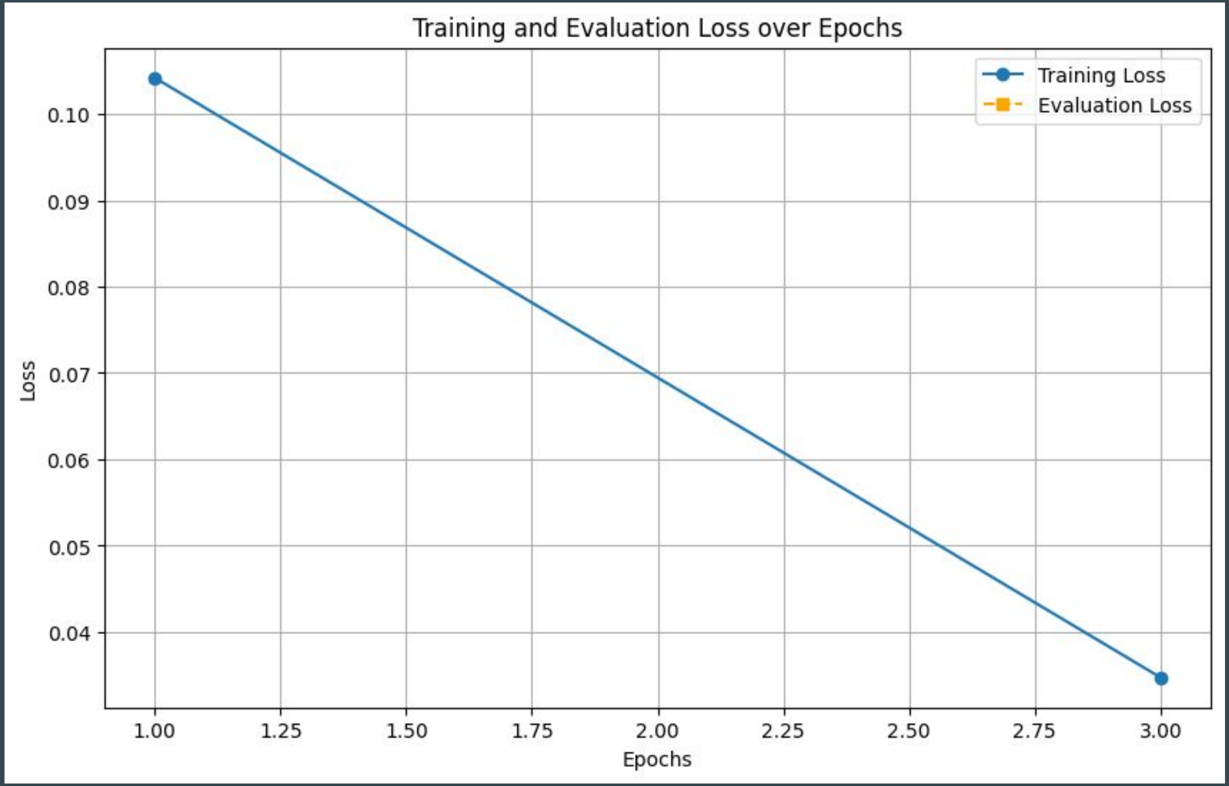


GPT2-XL ONLY OLYMPICS

Evaluation Timed out at 72 hours :(

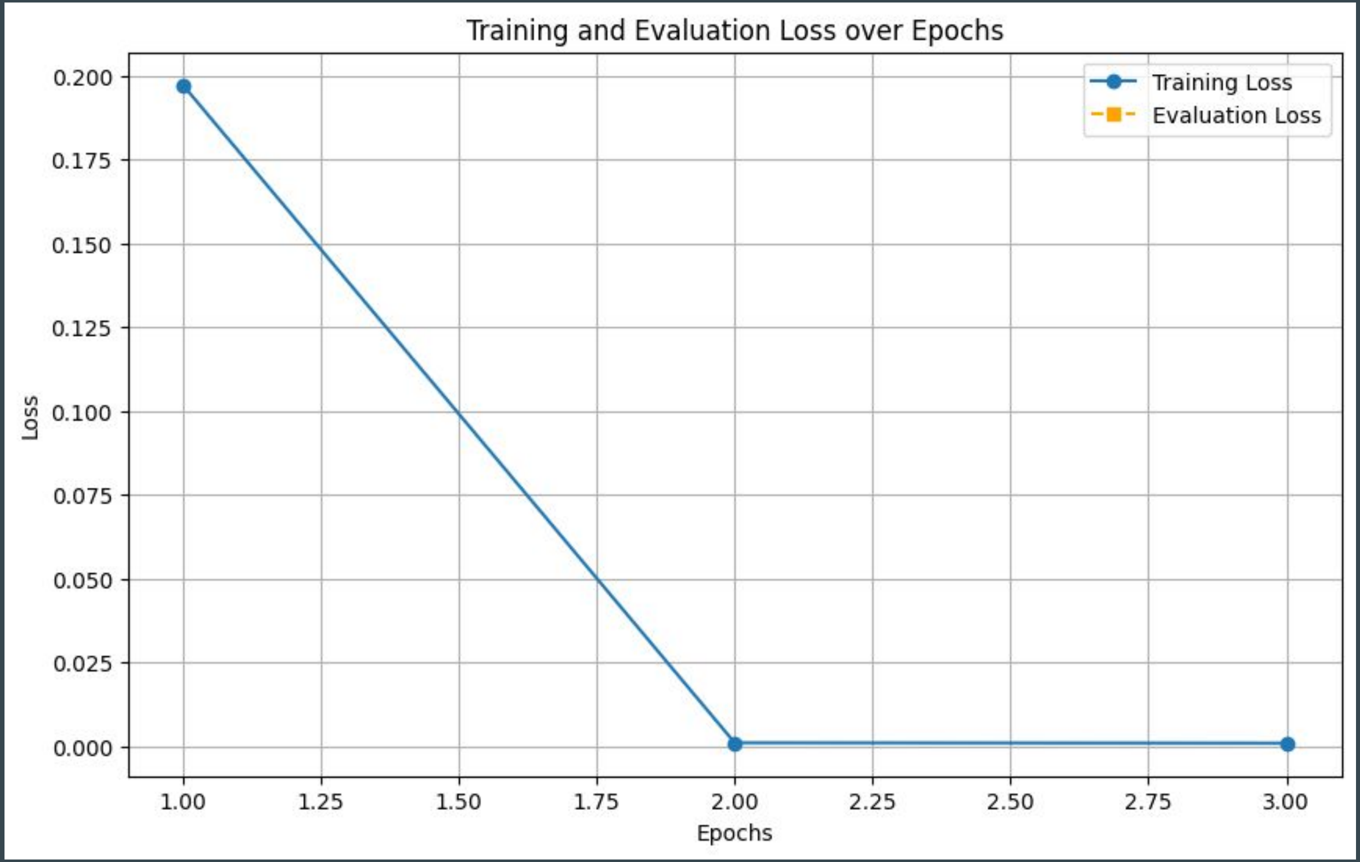


ALBERT



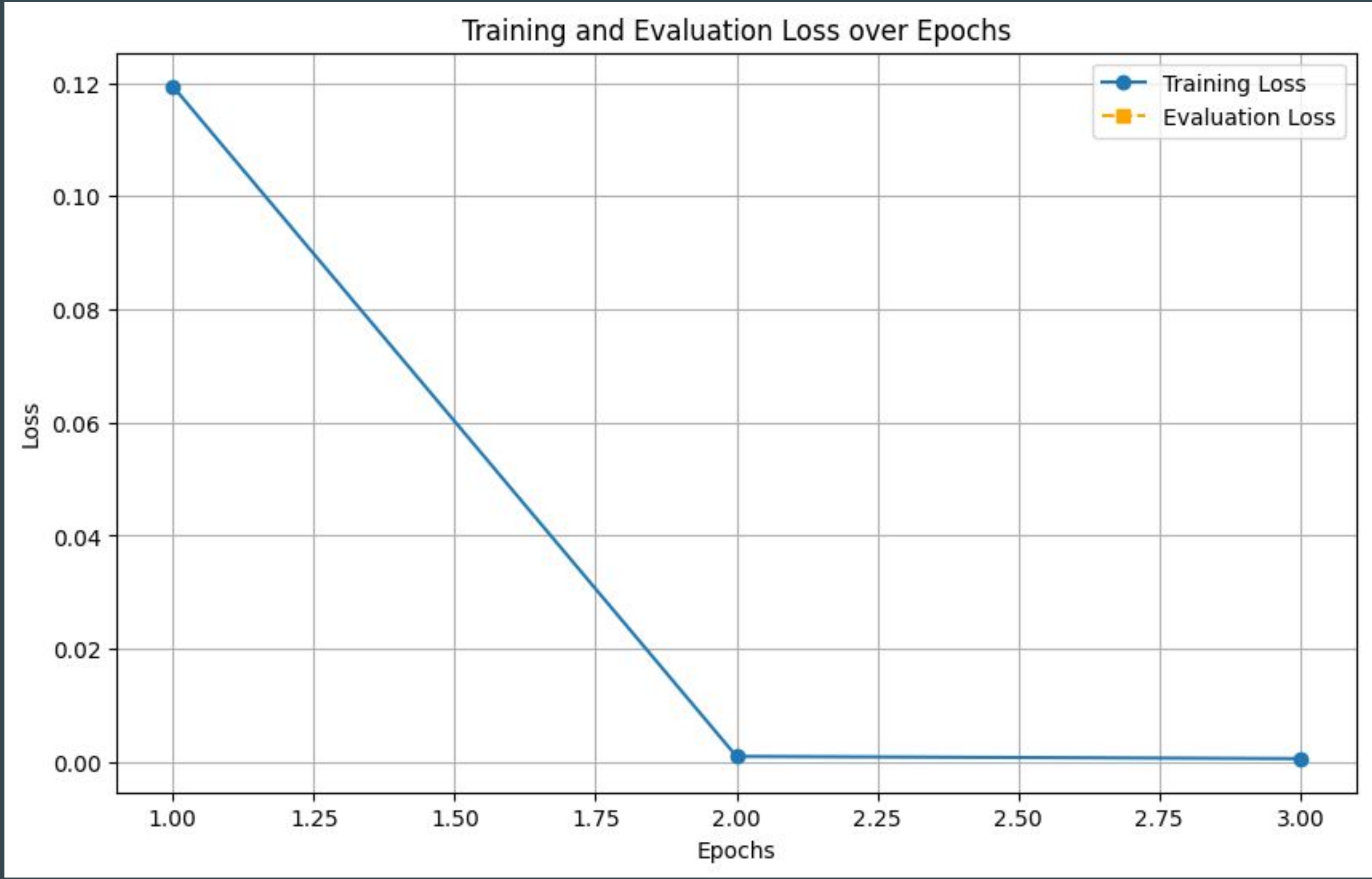
Training and Evaluation Loss over Epochs
Training Loss
Evaluation Loss
0.10
0.09
0.08
0.07
0.06
0.05
0.04
Loss
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Epochs

BERT



Training and Evaluation Loss over Epochs
Training Loss
Evaluation Loss
Loss
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Epochs

roBERTa

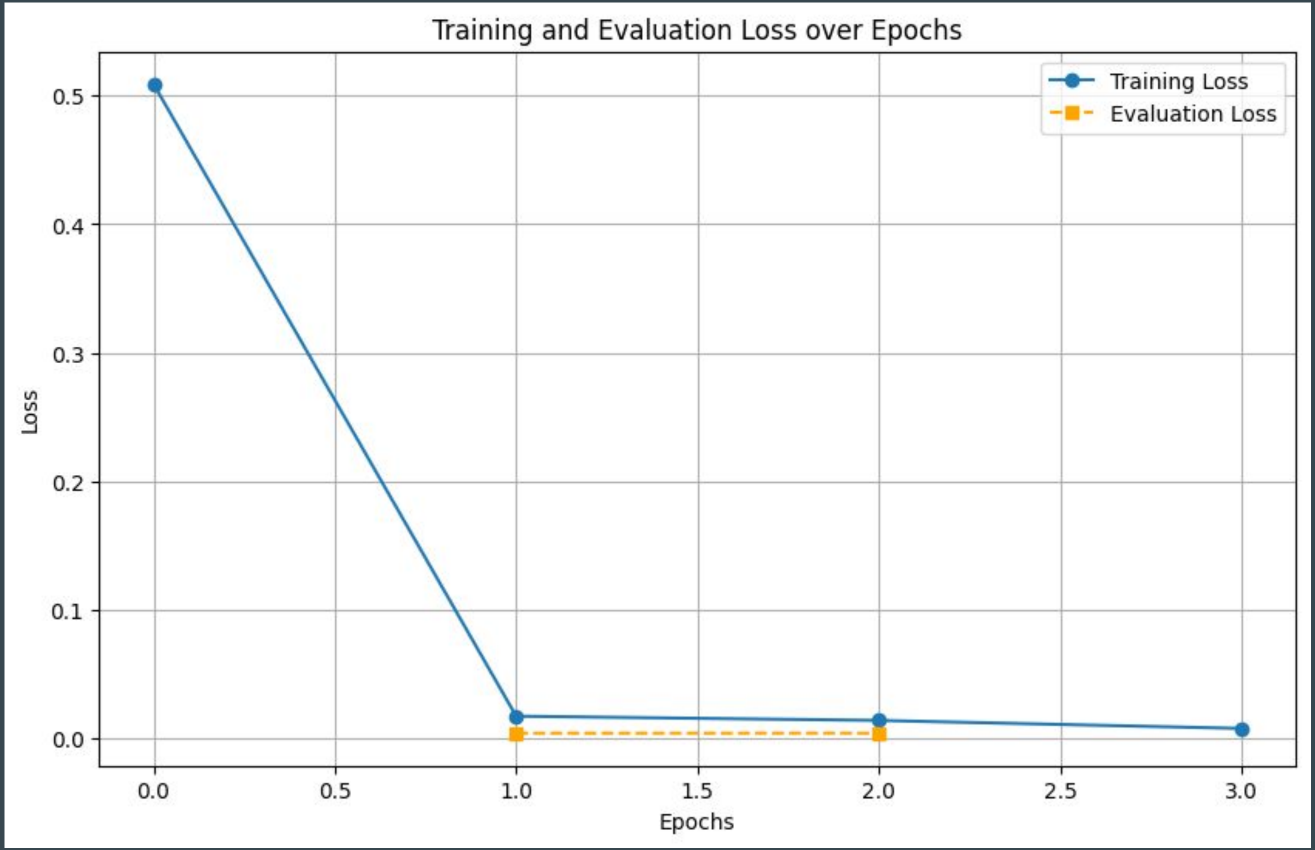


Training and Evaluation Loss over Epochs
Training Loss
Evaluation Loss
Loss
0.12
0.10
0.08
0.06
0.04
0.02
0.00
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Epochs

FLAN-T5

Eval 1:
BLEU Score: 5.4010
ROUGE Scores:
  ROUGE1: 0.3556
  ROUGE2: 0.2613
  ROUGEL: 0.3556
BERTScore: 0.8514

Eval 2:
BLEU Score: 80.1234
ROUGE Scores:
  ROUGE1: 0.9480
  ROUGE2: 0.6381
  ROUGEL: 0.9479
BERTScore: 0.9831



FLAN-T5 CONTINUED

Generated vs. Ground Truth Answers:

Run 1:

```
Example 1:
Generated: <pad> 42.0 <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
Actual: 42.0
--------------------------------------------------
Example 2:
Generated: <pad> 2001-07-14 <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
Actual: 2001-07-14
--------------------------------------------------
Example 3:
Generated: <pad> 4.0 <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
Actual: 4.0
--------------------------------------------------
Example 4:
Generated: <pad> modern pentathlon <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
Actual: modern pentathlon
--------------------------------------------------
```

Run 2:

```
Example 1:
Generated: 42.0
Actual: 42.0
--------------------------------------------------
Example 2:
Generated: 2001-07-14
Actual: 2001-07-14
--------------------------------------------------
Example 3:
Generated: 4.0
Actual: 4.0
--------------------------------------------------
Example 4:
Generated: modern pentathlon
Actual: modern pentathlon
--------------------------------------------------
```

RESULTS

Null Hypothesis: Tuned models (FLAN-T5) will not outperform the baseline model (gpt2-large).

Alternative hypothesis: The tuned models will outperform the baseline model.

Synthetic Value Generation - using model means and a variance of 0.02

HYPOTHESIS TESTING RESULTS:

	Mean Baseline	Mean Flan-T5	Variance Baseline	Variance Flan-T5
ROUGE1	0.035814	0.963109	0.016330	0.015476
ROUGE2	0.037354	0.630180	0.018008	0.022404
ROUGEL	0.059678	0.931593	0.023278	0.016888

	MSE	t-statistic	p-value
ROUGE1	0.897096	51.733951	2.760721e-117
ROUGE2	0.384190	29.341993	2.532560e-74
ROUGEL	0.806432	43.287470	3.349077e-103

Result Discussion

HYPOTHESIS TESTING RESULTS:

	Mean Baseline	Mean Flan-T5	Variance Baseline	Variance Flan-T5
ROUGE1	0.035814	0.963109	0.016330	0.015476
ROUGE2	0.037354	0.630180	0.018008	0.022404
ROUGEL	0.059678	0.931593	0.023278	0.016888

	MSE	t-statistic	p-value
ROUGE1	0.897096	51.733951	2.760721e-117
ROUGE2	0.384190	29.341993	2.532560e-74
ROUGEL	0.806432	43.287470	3.349077e-103

- Mean - FLAN-T5 outperforms baseline gpt2-large
  - ROUGE1, ROUGE2 (less pronounced), ROUGEL
    - ROUGE2 is expected to be lower than ROUGE1 and ROUGEL as it's stricter
      - requires two bigram matches - exact matches.
- Variance:
  - Baseline is higher, indicating more inconsistency compared to FLAN-T5.
- MSE:
  - Performance gap for ROUGE1 and ROUGEL is larger than for ROUGE2
- t-statistic:
  - High for all metrics - shows difference in means is very significant
- p-values:
  - All values are far below the significance threshold (0.05), and in fact effectively 0.
    - **We reject the null hypothesis** with extremely high confidence.

Most Challenging Aspects

Model Training / Evaluation - Computationally expensive

- Beocat jobs lasting a suspiciously long time
  - Not efficiently utilizing GPUs?
- Running out of time
  - Not enough time for resubmission
- Beocat down!
- Stuck in the queue

eval_gpt2-xl_olympics	55:25:55	batch.q	Running
squad_gpt2-large_olympics	00:00:00	batch.q,killable.q	Queued
eval_gpt2-large_olympics	00:00:00	batch.q,killable.q	Queued
flan-t5_olympics	94:47:19	killable.q	Running
albert_train_eval_olympics	01:28:39	killable.q	Running
just_bert_train_eval_olympics	02:02:08	killable.q	Running
roberta_train_eval_olympics	02:53:59	killable.q	Running

Challenges cont.

Learning how to train a QA chatbot

- Data formatting
- Process

Model Training

- Selecting / tuning hyperparameters
- Time limits, re-running

Setting up models correctly

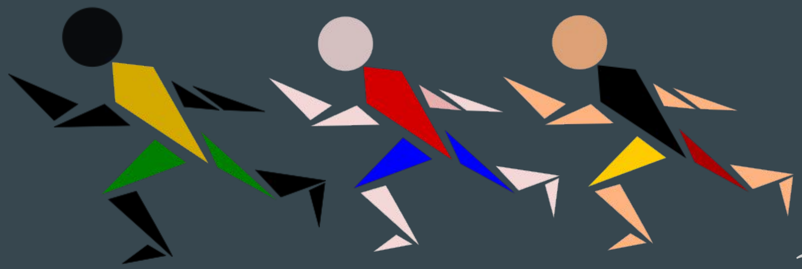
- Spent hours trying to get BERT models to work - testing on Beocat was time consuming
- Difficult to debug

Aspects of Most Learning

Transforming datasets into required structures for QA task (mapping, etc)

Learning the process of fine-tuning LLMs and training for a QA task

Running large programs on Beocat



Future Work

Dedicate more time to training and evaluating models

- I expected long train times, but not as long as they ended up being. It messed up my timing badly

Chatbot UI

Further data cleaning and organization

- Develop more QA pairs

CONCLUSIONS

FLAN-T5 produced the best results for this task

- This model was well suited for the Olympics QA dataset

roBERTa and gpt2-large both showed promising loss plots, further investigation would be beneficial

Indicates that for this QA task, transformers may be sufficient

- We could save time and memory by training a transformer instead of LLM

Thank you!
Questions?