Final Report

Project Title: Open-Domain Question Answering System with Hybrid Retrieval and Dual Answering Models
Team Members: Abbas Aliyev, Huru Algayeva

1. Motivation

We tackled the problem of Open-Domain Question Answering (ODQA), where the system must retrieve and generate answers from a large unstructured text corpus (e.g., SQuAD) without access to a predefined context. ODQA is fundamental in real-world NLP applications like chatbots, search engines, and AI assistants. Our goal was to build a modular and extensible QA system using both extractive and generative approaches, integrating multiple retrievers and answer generation methods.

2. Method

Dataset:

University_of_Notre_Dame	The Joan B. Kroc Institute for International Peace Studies at the University of Notre Dame is dedica...	In what year was the Joan B. Kroc Institute for International Peace Studies founded?	1986
University_of_Notre_Dame	The Joan B. Kroc Institute for International Peace Studies at the University of Notre Dame is dedica...	To whom was John B. Kroc married?	Ray Kroc
University_of_Notre_Dame	The Joan B. Kroc Institute for International Peace Studies at the University of Notre Dame is dedica...	What company did Ray Kroc own?	McDonald's
University_of_Notre_Dame	The library system of the university is divided between the main library and each of the colleges an	How many stories tall is the main library at Notre Dame?	14

A benchmark reading comprehension dataset with over 100,000 question-answer pairs on articles. Each question is answered by extracting a span of text from the given context. Used to train and evaluate extractive QA models.

Retrieval Modules:

- **Sparse Retrieval:** TF-IDF Vectorizer with cosine similarity.
- **Dense Retrieval:** Sentence-BERT (multi-qa-MiniLM-L6-cos-v1) for semantic similarity.

Answering Modules:

- **Extractive QA:** BERT-based QA model (bert-large-uncased-whole-word-masking-finetuned-squad).
- **Generative QA:** Sequence-to-sequence model (T5-base) for generating full answers from augmented contexts.

Preprocessing:

- BERT tokenizer and custom character vocabulary were generated to support downstream character-based CNNs.
- SpaCy was used to tokenize context and questions for robust answer span alignment.

3. Experiments

Experiment Setup:

- Dataset: SQuAD v1.1 (used a subset of 30,000 samples).
- We benchmarked both sparse and dense retrieval pipelines.
- For QA, we tested extractive vs. generative responses using retrieved documents.

Results:

- **TF-IDF Retriever + BiDAF (Extractive)** yielded good results on straightforward questions with high lexical overlap.
- **SBERT + T5 (Generative)** performed better on questions requiring rephrasing or synthesis, though it was more resource-intensive.
- Combined approach improved flexibility and robustness of the system.

```
query = "Who was the first president of the United States?"
answer = rag_pipeline(query, retriever, answer_mode='generate')
print("Answer:", answer)
```

```
Answer: George Washington
```

```
query = "What is the chemical symbol for hydrogen?"
answer = rag_pipeline(query, retriever, answer_mode='generate')
print("Answer:", answer)
```

```
Answer: H
```

Bidaf Implementation result:

```
        Train Loss: 1.2436
        Val Loss: 4.0594 | Val EM: 42.86% | Val F1: 59.38%

Training complete.
Best Validation F1: 60.65% (Model saved at ./bidaf_best_model.pt)
```

Bidaf Implementation with Bert Embedding result:

Epoch 4 results:

exact match - 52.21%

F1 score - 70.37%

Already better than **bidaf with glove.**

Observations:

- Dense retrievers captured semantic relationships better but were slower and more memory-hungry.
- Extractive QA failed when answer span wasn't clearly delineated in retrieved contexts.
- T5 was powerful but often verbose or hallucinated facts on less-relevant context slices.

Error Analysis:

- Misalignment between retrieved contexts and ground truth answers was the main bottleneck.
- Some questions had multiple plausible answers; metrics like EM and F1 underrepresented model performance in such cases.
- Answer span detection was particularly sensitive to tokenization mismatches in extractive models.

4. Contributions

Abbas Aliyev:

- Implemented preprocessing pipelines using HuggingFace Datasets and BERT tokenizer.
- Built custom character-level vocab for BiDAF-style architectures.
- Explored token alignment challenges using SpaCy and debugged span-matching logic.
- Ran error analysis on token mismatches and retrieval errors.

Huru Algayeva:

- Designed the full hybrid ODQA pipeline.
- Implemented and tested both Sparse (TF-IDF) and Dense (SBERT) retrievers.
- Developed generative and extractive QA modules.
- Tuned hyperparameters, optimized model inference, and led final evaluation and integration.

5. Conclusion

This project demonstrates the strengths and weaknesses of hybrid RAG (Retrieval-Augmented Generation) systems. While combining multiple retrievers and answerers boosts performance and flexibility, ensuring alignment between components remains a challenge. Future improvements could include better truncation logic, and dataset-specific fine-tuning.