Unveiling the Past: AI Powered Historical Book Question Answering System (HBQAS)

Hari Thapliyal,
MBA, MCA, MS (AI), PGD(Finance), PMP, PMI-ACP, PRINCE2

AGENDA

Problem Statement

How to validate if message has been received along with the intent?

- An story is narrated to someone but s/he understood is ….
- A question is asked but the answer is ….
- A story or a book is read but the moral of story is ….
- You visited some place have fun time but what you learnt….
- More can be added.

This applies to all…. personal, family, social, corporate, business and spiritual aspects of life.

One Solution: Have a reliable Question Answering System in Place

Project Objective

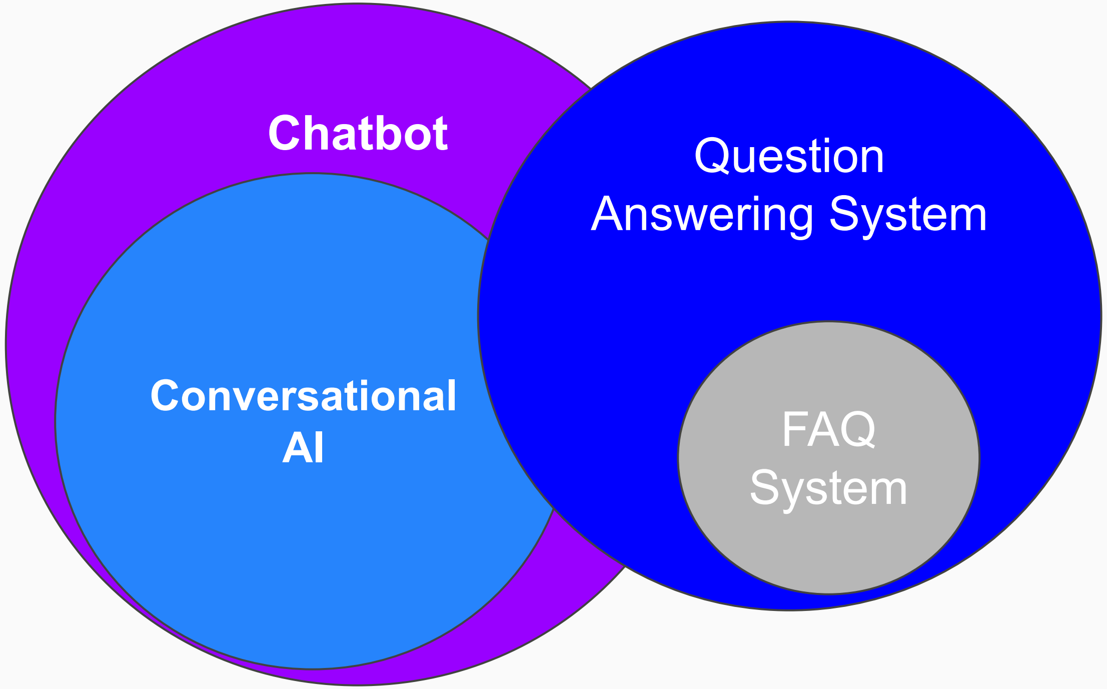
- Developing a System for Question, Answer Generation from any Historical book at the minimal cost. (Question Answer Generation System - QAGS)
- Developing answer generation system which search the answer in a given context/text. (Answer Generation System - AGS)
- Developing a system for searching a document, which has answer to a question. (Document Retrieval System - DRS)
- Developing a system for searching answer to a question without any context. (Retrieval Augmented Answer Generation System - RAAGS)

Research Questions

1. What is the cost-effective way of generating high-quality **question-answer dataset** from a historical text?
2. What is the cost-effective way of generating high-quality **answers** from historical text-based Questions?
3. What different NLP/NLU/NLG **technologies are available** for QA tasks in general?
4. Under 30 minutes, is it possible to generate **one thousand high-quality questions and answers** with reference from translated Mahabharata text using freely available hardware resources?
5. How to create a system that can **answer any question from a historical book** without any reference QA paired dataset?
6. What are the **linguistic nuances of Indian Historical text** that impact the quality of the translation of questions and answers between the Indian Language?

Introduction to QAS

QAS, CAI, Chatbot, FAQ Relationship



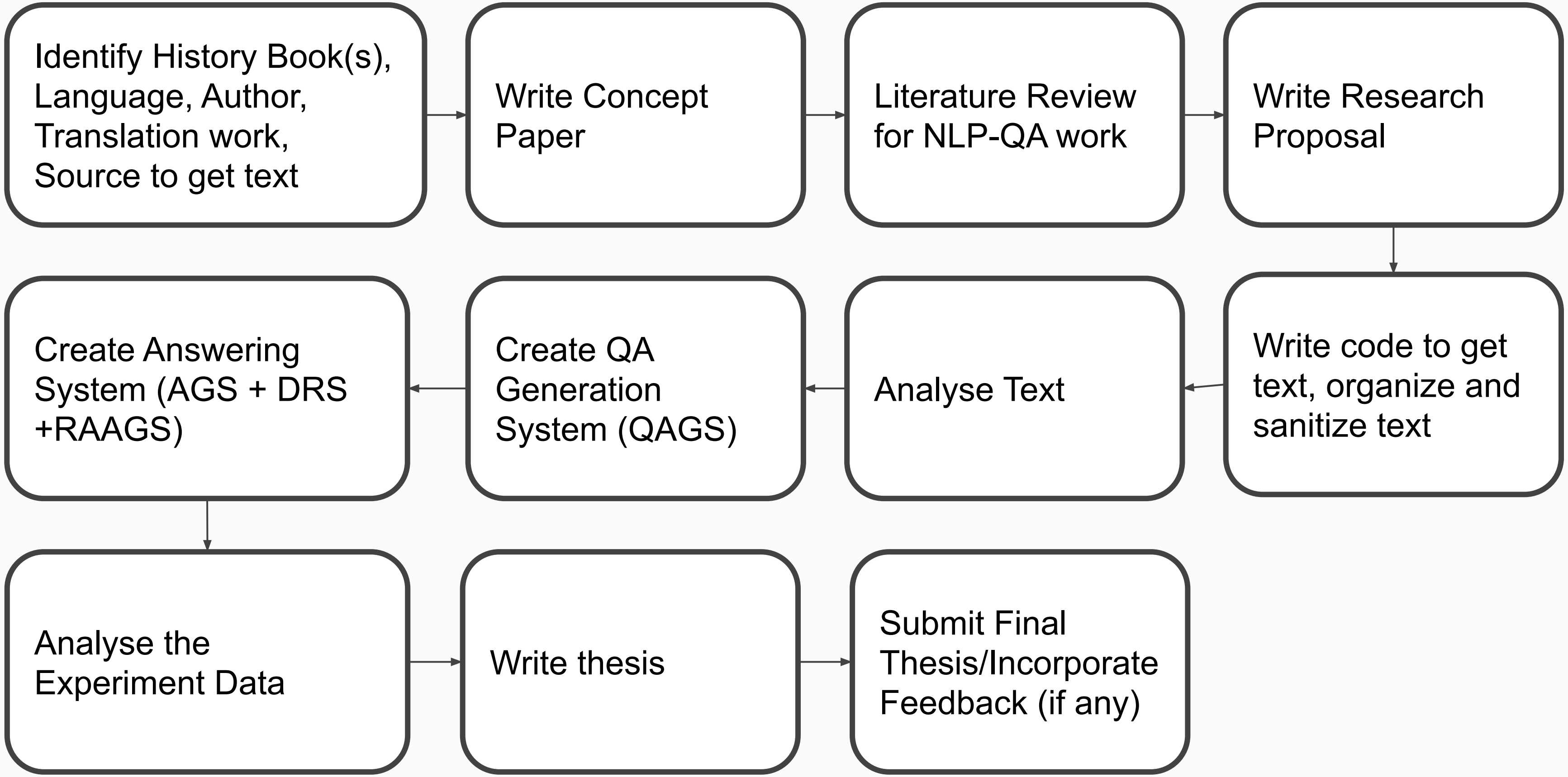
Introduction to QAS

Journey of QAS/Chatbots

ELIZA	1966
LUNAR	1971
RNN	1988
ASK	1996
Jabberwacky	1997
LSTM	1997
Google Search	1998
Siri	2011
Telegram	2013
IBM Watson	2013
Slack	2013
GRU	2014
Cortana	2014
Alexa	2014
Google Assistant	2016
Transformer	2017
	2017
Hundreds of Products	Onwards

Research Design Diagram

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.


Identify History Book(s), Language, Author, Translation work, Source to get text
Write Concept Paper
Literature Review for NLP-QA work
Write Research Proposal
Write code to get text, organize and sanitize text
Analyse Text
Create QA Generation System (QAGS)
Create Answering System (AGS + DRS +RAAGS)
Analyse the Experiment Data
Write thesis
Submit Final Thesis/Incorporate Feedback (if any)

Subsystems of HBQA

What do you want to know from a history book? Which system should be used at what level of awareness about historical book?

		I want to...		
		Be aware	Know the Document Name	Know the Answer
Awareness	I don't have question	QAGS	X	X
	I have question, but I don't know the context	X	DRS	RAAGS
	I have a question and context	X	DRS*	AGS

DRS* Use DRS without context.

HBQA System Evaluation

QAGS - Metrics	DRS - Metrics	AGS - Metrics	RAAGS - Metrics
1. Unique Questions 2. Question Answer Cosine 3. Question/1000 Words in Chunk 4. Answer/Question Length Ratio	1. P@n 2. R@n 3. F1@n 4. MAP 5. MRR 6. FoundIn@n	1. BLEU@n 2. ROUGE@n 3. Precision 4. Recall 5. Cosine	1. BLEU@n 2. ROUGE@n 3. Precision 4. Recall 5. MRR 6. Cosine

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.

1. QA dataset on Mahabharat. 1850+ questions in the English Language.
2. Explored approaches for Question Answer Generation
3. Explored approaches for Answer generation from a historical book.
4. Explored metrics that can help in evaluating this kind of work.
5. Explored sentence embedding models that can be used for Indian historical books translated into English language.
6. Evaluated different SOTA Question Answering Models for Historical text and finetuned models for our work.
7. Explored whether we can answer a question without any context.
8. Explored how efficient is document retrieval in the case of Indian Historical text
9. High dimension embedding does not mean the best sentence vector. We need to understand what kind of sentence embedding model is suitable for our work. Secondly, in the race for LLM which is expensive to train and deploy, we can still work with models like T5

The Contribution Project

Applications

1. We can **extend t**his work to any other **language** or any other historical **book**
2. It can help make historical texts **more accessible** and understandable to a wider audience, including researchers, students, and enthusiasts
3. AI and NLP are evolving fast and history is becoming more relevant to solve the current struggle, **to understand the background**, and to understand the value system of societies
4. Our research could **bridge the gap** between historical texts and how modern people **understand that today**.
5. **Archaeology sites**
6. During **Academic Class**: Based on the content delivered last day, last week, last semester.
7. Creating **Question-Answer Books**

Applications

8. **Other Question Formats**: We have explored open-ended answers but we can take this work forward for other formats like fill-in-the-blank, and MCQ

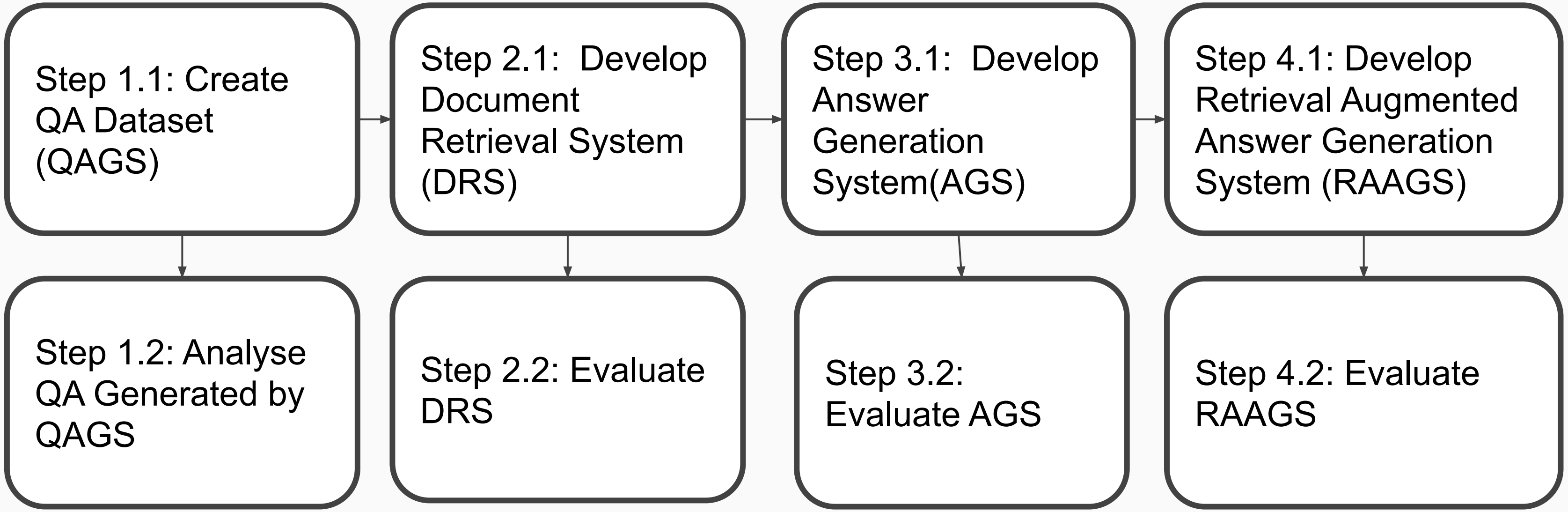
9. **Live TV Shows**: Generally television hot debate to understand the history and perspective of people sucks lots of energy and makes people bitter. An anchor can use QAGS, AGS, RAAGS to drive the debate in a more meaningful way.

10. **Quiz in family, Group**: People are turning towards their phones and TV, and human-to-human connection is reducing. We can discuss the topic with family members and create a quiz that they can use to play with each other in a family setup.

11. The QA dataset created in this project can be **translated into other Indian Languages**

12. **We can extend this work for other non-history domains as well.**

HBQA System High Level Approach


Step 1.1: Create QA Dataset (QAGS)
Step 2.1: Develop Document Retrieval System (DRS)
Step 3.1: Develop Answer Generation System(AGS)
Step 4.1: Develop Retrieval Augmented Answer Generation System (RAAGS)
Step 1.2: Analyse QA Generated by QAGS
Step 2.2: Evaluate DRS
Step 3.2: Evaluate AGS
Step 4.2: Evaluate RAAGS

High Level Approach and Deliverables

- **QA Dataset Creation**: Using ChatGPT or other Alternative software Generate Question Answer from a Historical Book.

  Deliverable: 1000+ QA Dataset on a Historical Book called Mahabharata
- **Model Development:** Explore Zeroshot models, Developing multiple fine-tuned model for answer generation

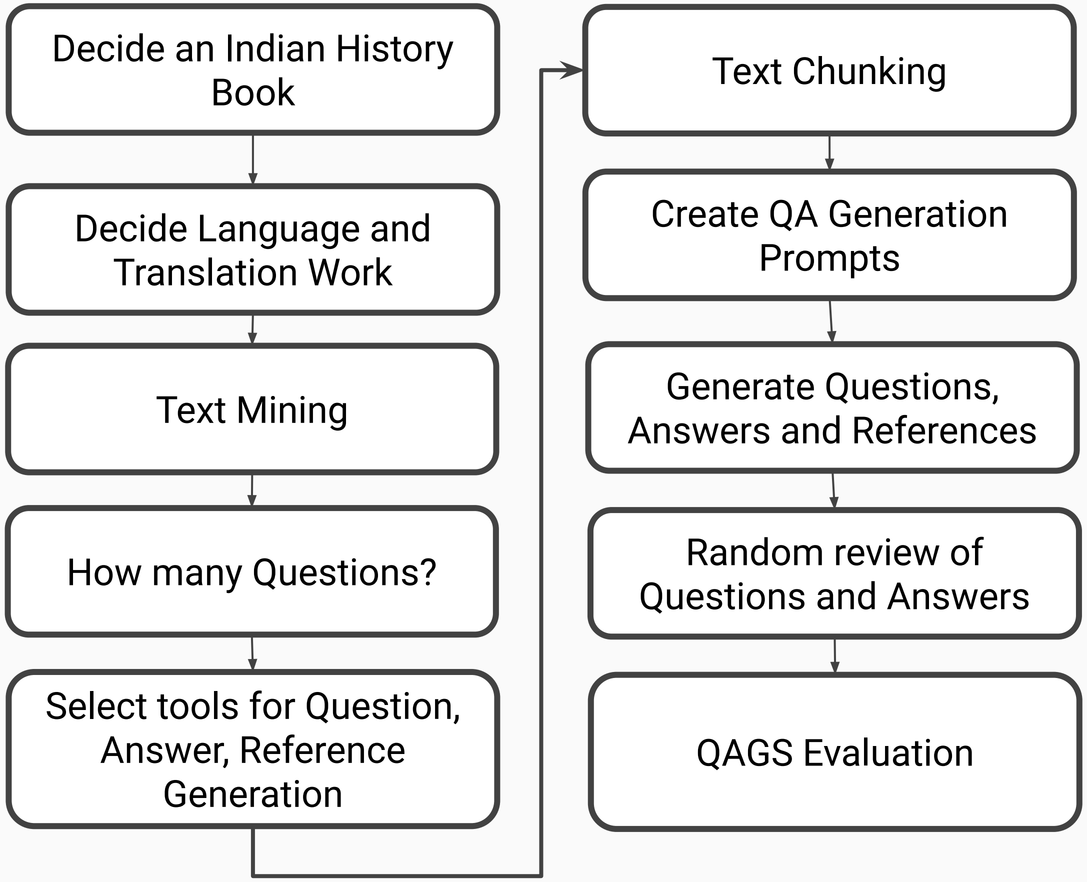
  Deliverable: The best answer generating finetuned model.
- **Model Evaluation** : Compare Model Generated answer to Reference Answer.

  Deliverable: Performance of model on various quality metrics.
- **Search Answer** : Search Answer of Question in the absence of context

  Deliverable: An approach, how to search answer of question from a big corpus of many books.

Question Answer Generation System (QAGS)

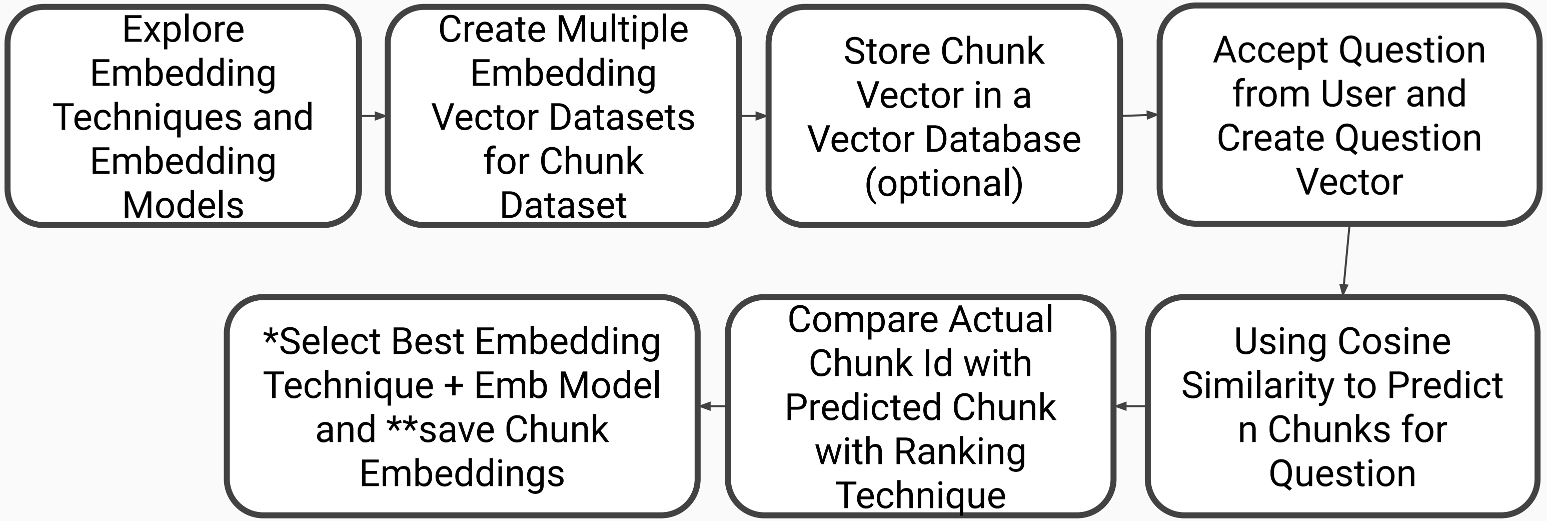
1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.


Decide an Indian History Book
Decide Language and Translation Work
Text Mining
How many Questions?
Select tools for Question, Answer, Reference Generation
Text Chunking
Create QA Generation Prompts
Generate Questions, Answers and References
Random review of Questions and Answers
QAGS Evaluation

Building Question Answer Generation System (QAGS)

1. **Decide an Indian History Book** : The Mahabharata Selected
2. **Decide Language and Translator** : English Language Translation of KM Ganguli Selected.
3. **Text Mining**: Web Scraping of 18 Parvas, 2100 Sections: Download 2100 text files of total 14MB Size.
4. **How many Questions?**: Decide how many/ what kinds of questions needed. Select chapter to create Questions from. Selected Book 3, all the section of this Parva are taken, apart from this smaller section of Parva 1, 2,4,5, 7 also taken. We need minimum 1000 descriptive QA.
5. **Select tools for Question, Answer, Reference Generation**: Decide what tool to use for creating questions, answers and references. Decided to use ChatGPT, ChatGPT API, ChatPDF.
6. **Text Chunking**: Used Langchain Library for splitting the Text.
7. **Create Question Prompts**: Create Queries (ChatGPT Prompts) for QA. Created a Prompt dataset.
8. **Generate Questions, Answers and References**: Use selected tools to Create Questions, Answer and References from earlier created Prompts. Mahabharata-HBQA Dataset Created.
9. **Random review of Questions and Answers**: Remove duplicate, ambiguous, contextless, meaningless questions. Mahabharata-HBQA Dataset Updated.
10. **QAGS Evaluation :** Using cosine to measure the distance between Context, Question & Answer.
11. **Train Test Split**: Create Train and Test split on Mahabharata-HBQA Dataset

Document Retrieval System (DRS)



Explore Embedding Techniques and Embedding Models
Create Multiple Embedding Vector Datasets for Chunk Dataset
Store Chunk Vector in a Vector Database (optional)
Accept Question from User and Create Question Vector
Using Cosine Similarity to Predict n Chunks for Question
Compare Actual Chunk Id with Predicted Chunk with Ranking Technique
*Select Best Embedding Technique + Emb Model and **save Chunk Embeddings

*In RAAGS, we will use this embedding model/technique to embed the question and search in chunk database.
*In AGS, we will also use this embedding to embed the predicted answer and reference answer. This helps us understanding the difference between predicted answer and reference answer.
**In the production environment use Vector Database like Pinecone to store this.

Building Document Retriever Model (DRS)

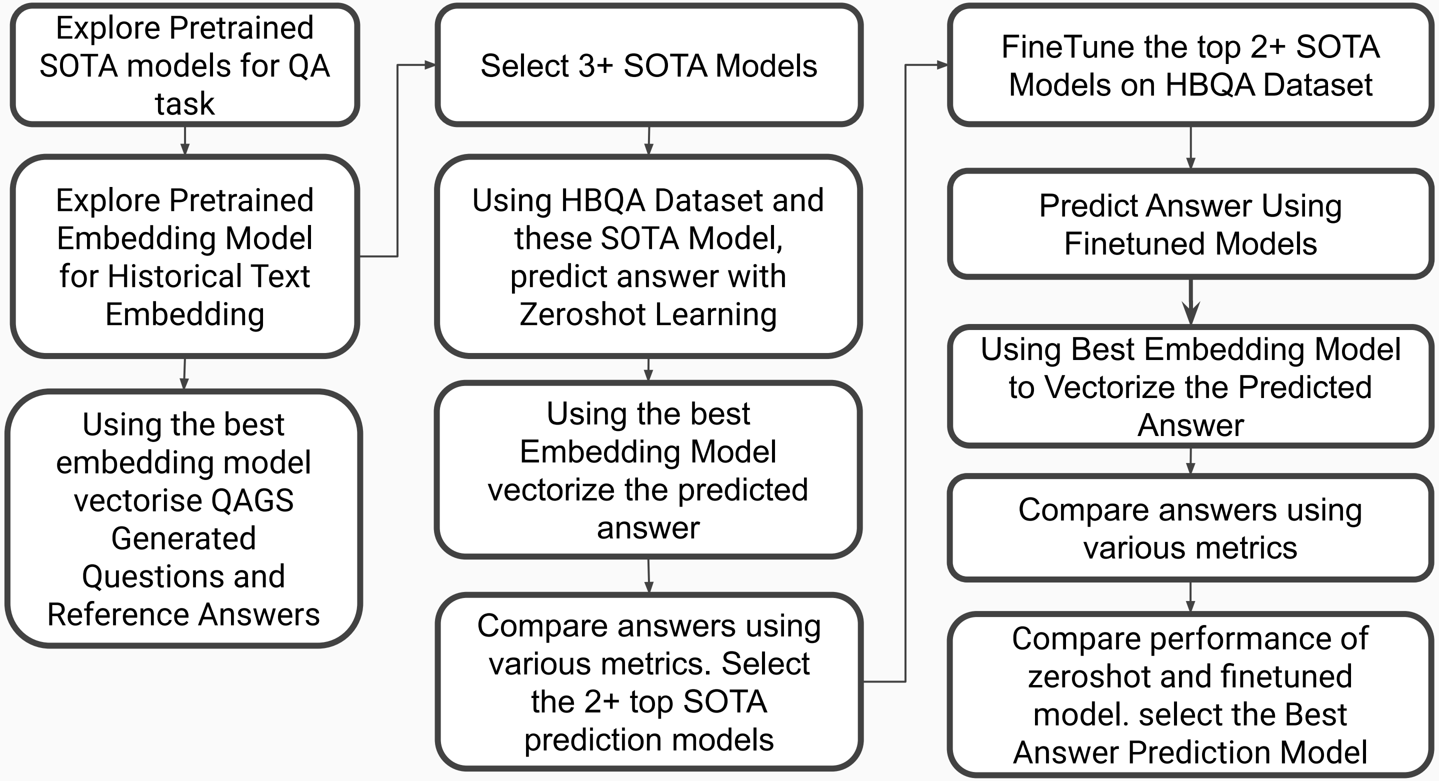
1. **Explore Word and Sentence Embedding Models:** Selected SentenceTransformer architecture and explored following embedding (1) multi-qa-distilbert-cos-v1, (2) all-MiniLM-L6-v2, (3) all-distilroberta-v1, (4) all-mpnet-base-v2, (5) all-roberta-large-v1
2. SentenceTransformer has dozens of Embedding Model, they are different in size, dimension, normality, corpus used to create embedding.
3. **Create Multiple Embedding Vector Datasets**: Using multiple embedding models, **create multiple Vector Embeddings for Question, Reference Answer, and Chunk (Context).**
4. **Predict Chunk for Question**: Using **cosine similarity** between question vector and chunk vector select top 10 chunks for a given question.
5. **Select the best Embedding Model:** Compared the Predicted Chunk Id and Actual Chunk Id and **select the embedding "all-mpnet-base-v2" model.**

Embedding Models

These five embedding models are based on SentenceTransformer architecture.

#	Sentence Embedding	Data Source	Size	Max Seq Length	Dimension	Normalized Embeddings
1	multi-qa-distilbert-cos-v1	215M QA pairs from diverse sources.	250 MB	512	768	True
2	all-MiniLM-L6-v2	1B+ training pairs.	80 MB	256	384	True
3	all-distilroberta-v1	1B+ training pairs	290 MB	512	768	True
4	all-mpnet-base-v2	1B+ training pairs	420 MB	384	768	True
5	all-roberta-large-v1	1B+ training pairs	1.36 GB	256	1024	True

Answer Generation System (AGS)



Explore Pretrained SOTA models for QA task
Explore Pretrained Embedding Model for Historical Text Embedding
Using the best embedding model vectorise QAGS Generated Questions and Reference Answers
Select 3+ SOTA Models
Using HBQA Dataset and these SOTA Model, predict answer with Zeroshot Learning
Using the best Embedding Model vectorize the predicted answer
Compare answers using various metrics. Select the 2+ top SOTA prediction models
FineTune the top 2+ SOTA Models on HBQA Dataset
Predict Answer Using Finetuned Models
Using Best Embedding Model to Vectorize the Predicted Answer
Compare answers using various metrics
Compare performance of zeroshot and finetuned model. select the Best Answer Prediction Model

Building Answer Generation System (AGS)

1. **Identify SOTA models for QA tasks:** Selected T5, BERT, DistilBERT, FlanT5, RoBERTa, GPT2, Bloom models
2. **Identify Embedding Model** : Used same embedding which was used in DRS.
3. **Experiment Zeroshot Learning**: Using QAGS generated Questions and Chunks predict answer with SOTA models (without any finetuning)
4. **Evaluate Zeroshot Model**: Using BLEU score, ROUGE score, Precision, Recall metrics computed the quality of predicted answer.
5. **Select SOTA model for Finetuning**: Selected T5 and Flan-T5.
6. **FineTune each SOTA Model**: Each selected models is finetuned on QAGS generated Dataset.
7. **Generate Answer**: Generate each question's answer with Every FineTuned Model
8. **Create Vector Embedding** : Create embedding for each Predicted answer using Embedding model.
9. **Compute cosine** distance between predicted answer and reference answer
10. **Compute Answer Correctness**: Using BLEU score, ROUGE score, Precision, Recall, Cosine metrics computed the quality of predicted answer.
11. **Compare Model Performance**: Compare model performance against BLEU, ROUGE, Precision, Recall, Cosine metrics and select the the answer generation model.

Answer Prediction Models

These transformer based models are used in our work.

For Zeroshot

1. T5 and Flan-T5: Sequence-to-sequence models
2. DistilBERT and RoBERTa: Encoder-only models
3. BERTSquad: Encoder-only model specifically designed for question answering tasks
4. BigBird: Sequence-to-sequence model with long-range dependencies
5. BLOOM: Decoder-only model

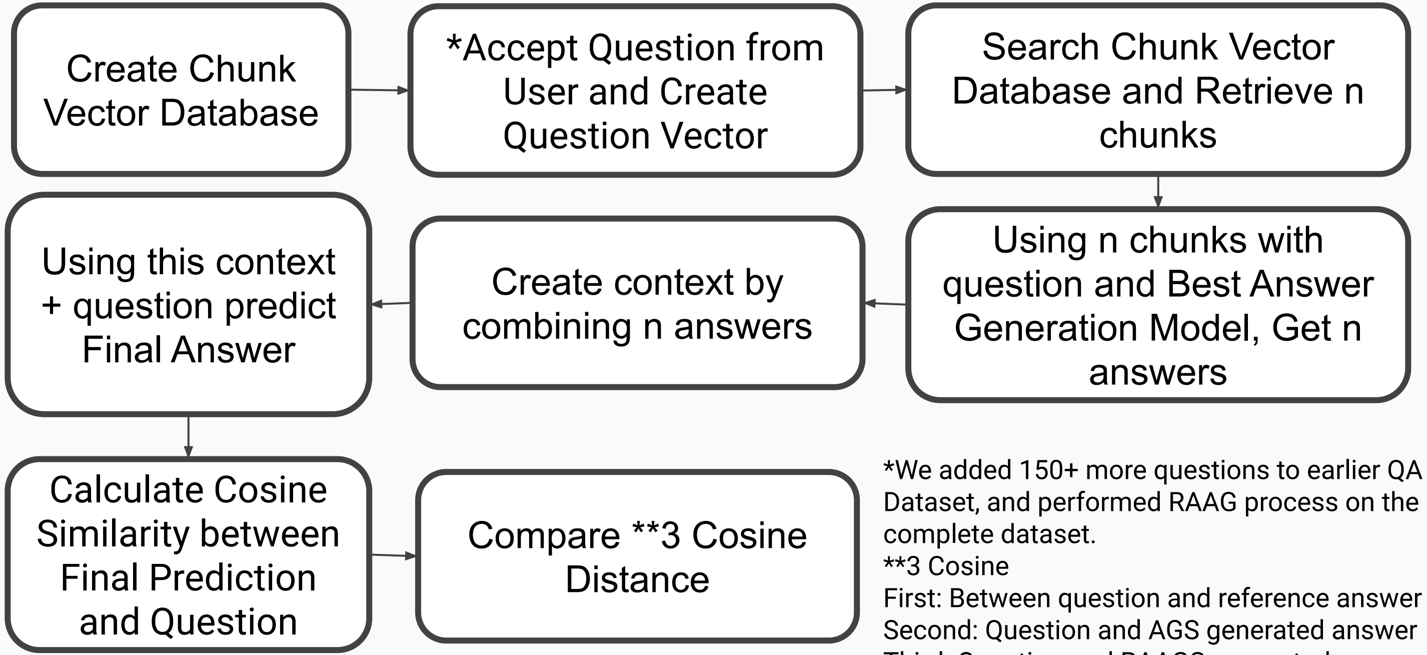
For Finetuning

1. T5 or Text-to-Text Transfer Transformer, is a sequence-to-sequence transformer language model developed by Google AI.
2. Flan-T5 : Flan (Finetuned Language) T5 is finetuned version of T5 from google. It was pre-trained on a massive dataset of text and code, and then fine-tuned on a mixture of tasks, including question answering, summarization, and translation.

What are three types of Transformers

- **Sequence-to-sequence models**: These models are designed to take a sequence of input tokens and generate a sequence of output tokens. They are commonly used for tasks such as generative question answering, machine translation and text summarization. Example: T5, FlanT5, BART
- **Encoder-only models**: These models are designed to encode a sequence of input tokens into a representation that can be used for downstream tasks such as classification and extractive question answering. They read sentence from both the direction. Example: BERT, RoBERTa.
- **Decoder-only/ Autoregressive models**: These models are designed to generate a sequence of output tokens from a latent representation. They are commonly used for tasks such as text generation and code generation. They read sentence from only one direction. Example GPT

Retrieval Augmented Answer Generation System (RAAGS)



Create Chunk Vector Database
*Accept Question from User and Create Question Vector
Search Chunk Vector Database and Retrieve n chunks
Using n chunks with question and Best Answer Generation Model, Get n answers
Create context by combining n answers
Using this context + question predict Final Answer
Calculate Cosine Similarity between Final Prediction and Question
Compare **3 Cosine Distance

*We added 150+ more questions to earlier QA Dataset, and performed RAAG process on the complete dataset.
**3 Cosine
First: Between question and reference answer
Second: Question and AGS generated answer
Third: Question and RAAGS generated answer

Retrieval Augmented Answer Generation System (RAAGS)

1. **Create Chunk Vector Database** : Chunk Vector Database using SentenceTransformer and all-mpnet-base-v2 embedding model
2. **Accept Question from User and Create Question Vector**: Create embedding vector using all-mpnet-base-v2 embedding model.
3. **Search Chunk Vector Database and Retrieve n chunks**: n=5, Five chunks were predicted for every question.
4. **Using n Chunks with Question and Best Answer Generation Model Get n Answers:** T5 Finetuned Model which was finetuned over 30 epoch was used.
5. **Create context by combining n answers**: Chunk corresponding to predicted chunk id is concatenated in the same order they were predicted.
6. **Using this context + question predict Final Answer**: Question along with newly created chunk were given to T5 finetuned model to predict the answer. It gives us final answer.
7. **Calculate Cosine Similarity** between Final Prediction and Question
8. **Compare *3 Cosine Distance**
    a. First: Between question and reference answer
    b. Second: Question and AGS generated answer
    c. Third: Question and RAAGS generated answer

Review RQ, Conclusion & Future

Answers to RQ

RQ1: What is the cost-effective way of generating high-quality questions from a historical text?

RQ4: Under 30 minutes, is it possible to generate one thousand high-quality questions and answers with reference from translated Mahabharata text using freely available hardware resources?

If we don't want to spend 17 hours on OpenAI front end to generate 1000 questions then we can use ChatGPT API and within 30 min we get 1000+ question answers. But we need to keep in mind, that it generates duplicate questions and secondly, we need to pay for the API. Finally, even if you execute the wrong prompt at openai API, you have to pay for that. Otherwise we can generate 1000+ good quality questions, answers and reference within 17 hours or 2 working days.

Answers to RQ

RQ2: What is the cost-effective way of generating high-quality answers from historical text-based Questions?

Fine-tuning an LLM model is time-consuming and expensive. But, we can finetune T5, FlanT5 transformer models with more epochs on free or almost free GPU machines. For historical question-answering work, we can get the best model that can answer all the questions of the given book.

Answers to RQ

RQ3: What different NLP/NLU/NLG technologies are available for Question Answering tasks in general?

We need to decide what kind of question-answer we want to generate. For MCQ and Extractive kind of QA system, almost all transformers can perform this task. For MCQ, we need to create a dataset with multiple options (for training and validation). For extractive QA, creating a suitable dataset is a laborious process because you need to mark the start and end of the answer in the chunk. In our case, we were interested in generative answers so we need not do these steps. And ChatGPT is good at creating generative question answers.

Answers to RQ

RQ5: How to create a system that can answer any question from a historical book without any reference question answer dataset?

Using RAAG system, we demonstrated that this is possible and we can generate good quality answers.

Note: We need a QA dataset (with limited number of QA pairs), for finetuning the model. But that model along with DRS can answer any question from the corpus.

Answers to RQ

RQ6: What are the linguistic nuances of Indian Historical text that impact the quality of the translation of questions and answers between the Indian Language?

This question was not related to experiments but analysis of Indian Historical text. If we create a QA dataset and QA system for the Mahabharat book in the English language and we want to translate these questions into Hindi or Tamil then it will generate lots of spelling errors around the noun. Secondly, we cannot ask questions in Devanagari or Tamil script in this system. Thirdly, the answer generated will always be in English language Latin script. Therefore, we recommend some steps to overcome this problem.

Future Recommendations

1. We need to understand to **pick up the most authentic translation** for this kind of work. Or generate multiple answers and mention as per which book what answer is generated. This helps people learn different viewpoints
2. Perform AGS experiments on LLM with **more parameters and bigger size models**.
3. Perform AGS experiments with **more epoch**
4. In QAGS, **generate multiple answers** to generated questions. Compare the predicted answer against all the alternative answers.
5. In DRS, create multiple high-dimensional **sentence embeddings for Indian Language historical text** which is translated into English. And use these embedding in DRS.
6. Create **QA in Indian languages**. We can translate, transliterate and transcribe QA generated using English language and Latin script into Indian Languages and Indian Scripts.
7. Create a **sentence embedding model for Indian script** (Devanagari, Kannada,Tamil) books.

Future Recommendations

8. Create a **model to predict the correct Devanagari spelling** of Roman script Indian Culture name entities like names of people, places, rituals, festivals, etc. Use these spellings during the translation process.
9. In AGS, to calculate BLEU and ROUGE scores, **keep only NER** in reference and predict the answers, BLEU and ROUGE scores may be better.
10. In DRS, **take multiple books on multiple topics** of the same domain say history.
11. Create **smaller chunks** to create chunk vectors and QAGS work to understand how DRS and other subsystems work.
12. In RAAGS, accept **questions from multiple books** and multiple topics.
13. In RAAGS, instead of joining multiple answers, **stuff the original chunks** using langchain and pass that final chunk to AGS, to get the answer.
14. **Human-in-the-Loop Evaluation**: Given the unique nature of the historical text, involving domain experts or historians in the evaluation process could provide valuable insights and enhance the quality of the evaluation. And this become more relevant when we are doing this work around a controversial history work

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.

METRICS

Complete Corpus Analysis

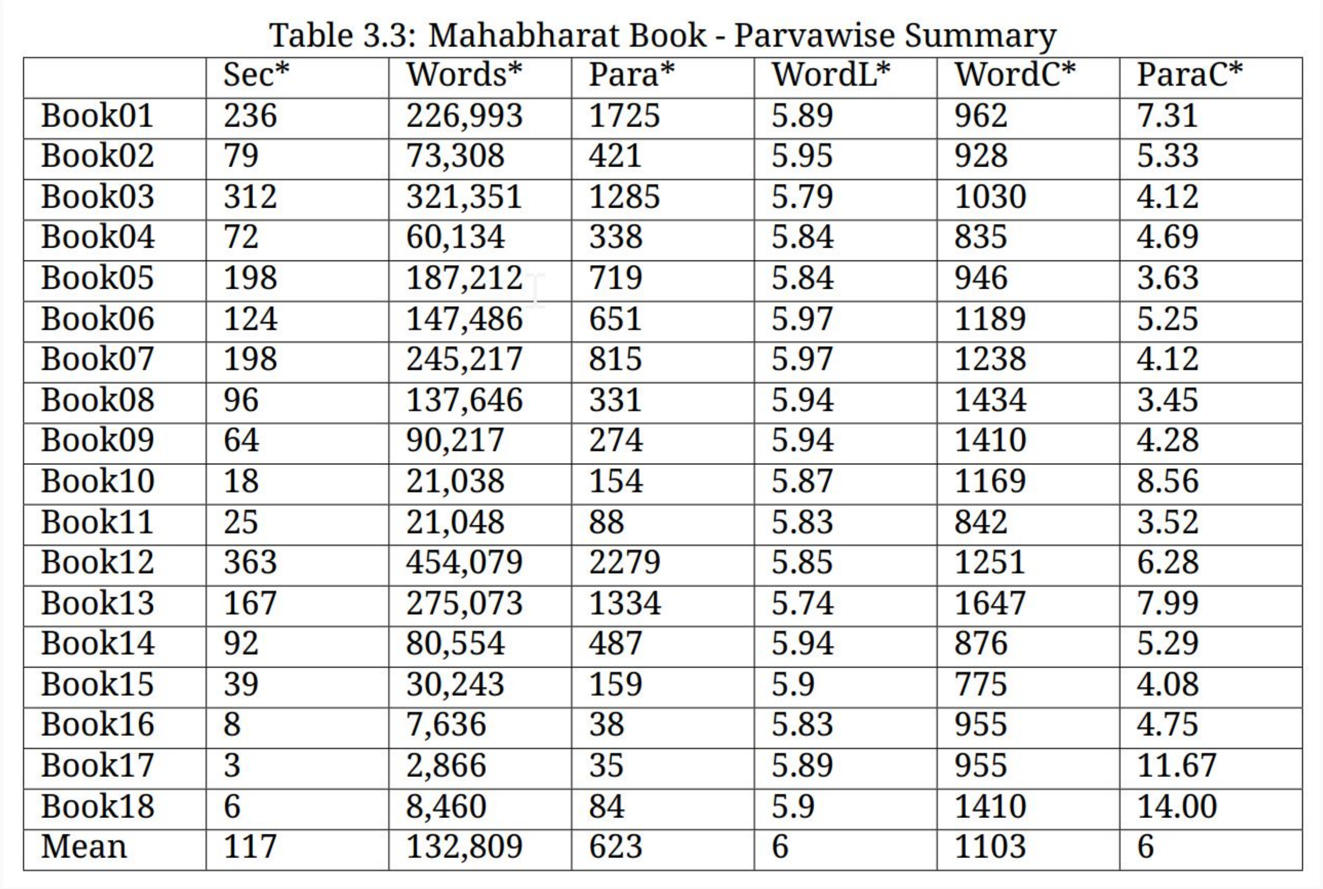


Table 3.3: Mahabharat Book - Parvawise Summary

	Sec*	Words*	Para*	WordL*	WordC*	ParaC*
Book01	236	226,993	1725	5.89	962	7.31
Book02	79	73,308	421	5.95	928	5.33
Book03	312	321,351	1285	5.79	1030	4.12
Book04	72	60,134	338	5.84	835	4.69
Book05	198	187,212	719	5.84	946	3.63
Book06	124	147,486	651	5.97	1189	5.25
Book07	198	245,217	815	5.97	1238	4.12
Book08	96	137,646	331	5.94	1434	3.45
Book09	64	90,217	274	5.94	1410	4.28
Book10	18	21,038	154	5.87	1169	8.56
Book11	25	21,048	88	5.83	842	3.52
Book12	363	454,079	2279	5.85	1251	6.28
Book13	167	275,073	1334	5.74	1647	7.99
Book14	92	80,554	487	5.94	876	5.29
Book15	39	30,243	159	5.9	775	4.08
Book16	8	7,636	38	5.83	955	4.75
Book17	3	2,866	35	5.89	955	11.67
Book18	6	8,460	84	5.9	1410	14.00
Mean	117	132,809	623	6	1103	6

Corpus Used Analysis

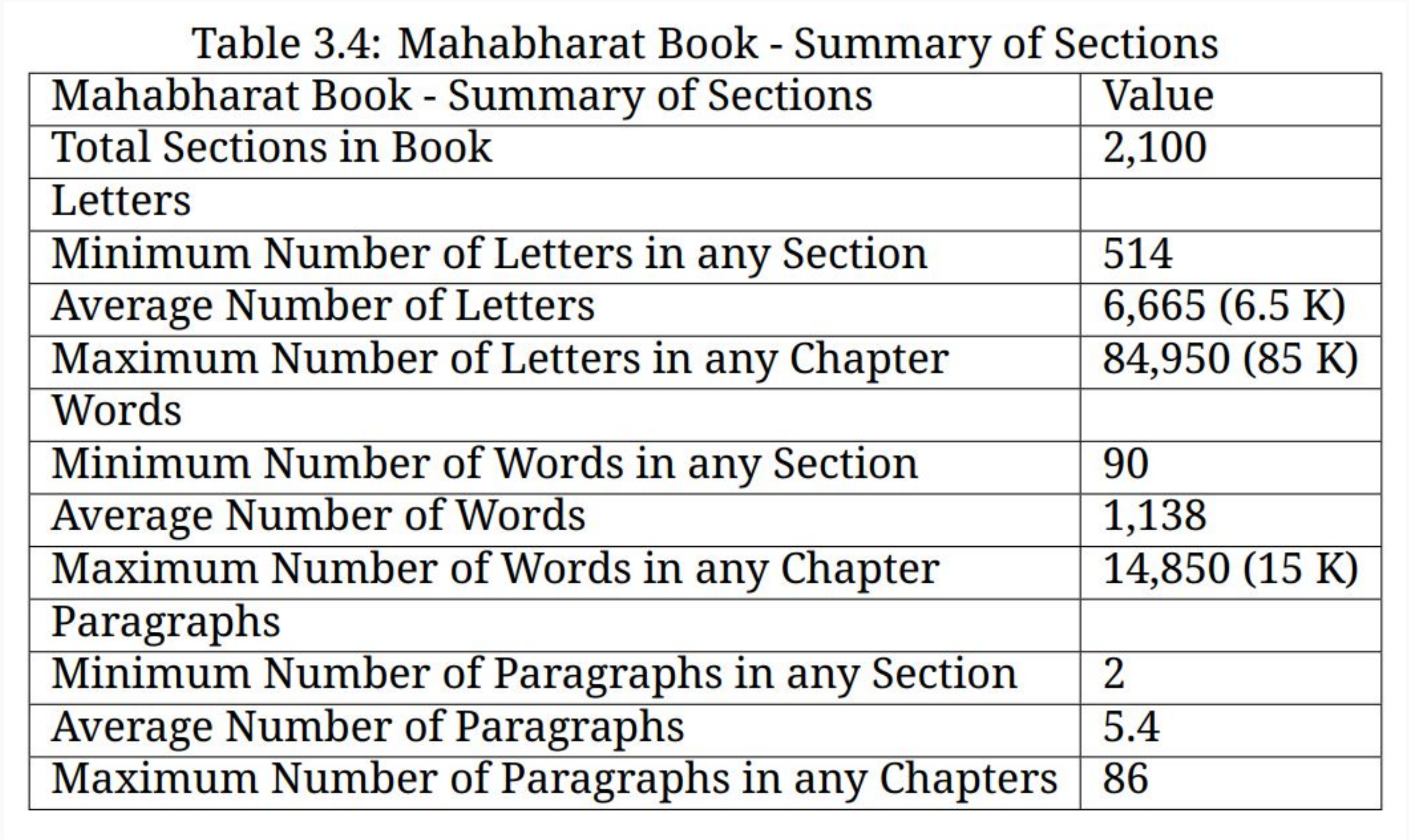


Table 3.4: Mahabharat Book - Summary of Sections

Mahabharat Book - Summary of Sections	Value
Total Sections in Book	2,100
Letters	
Minimum Number of Letters in any Section	514
Average Number of Letters	6,665 (6.5 K)
Maximum Number of Letters in any Chapter	84,950 (85 K)
Words	
Minimum Number of Words in any Section	90
Average Number of Words	1,138
Maximum Number of Words in any Chapter	14,850 (15 K)
Paragraphs	
Minimum Number of Paragraphs in any Section	2
Average Number of Paragraphs	5.4
Maximum Number of Paragraphs in any Chapters	86

Embedding Size

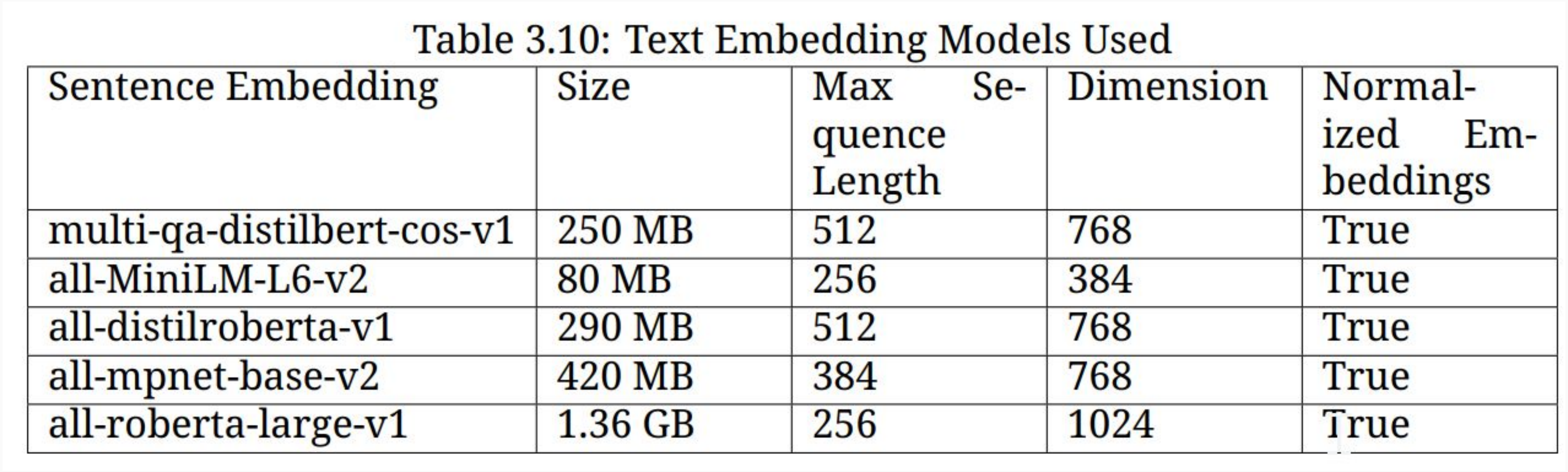
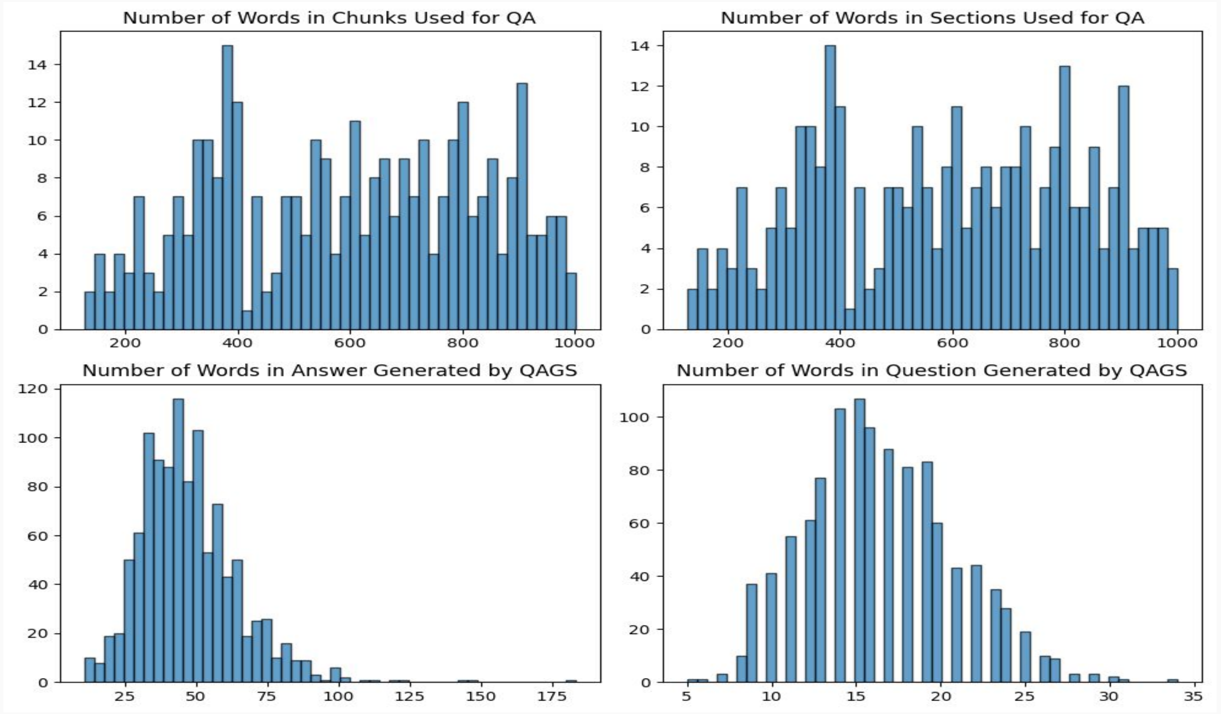


Table 3.10: Text Embedding Models Used

Sentence Embedding	Size	Max Sequence Length	Dimension	Normalized Embeddings
multi-qa-distilbert-cos-v1	250 MB	512	768	True
all-MiniLM-L6-v2	80 MB	256	384	True
all-distilroberta-v1	290 MB	512	768	True
all-mpnet-base-v2	420 MB	384	768	True
all-roberta-large-v1	1.36 GB	256	1024	True

QAGS Metrics

On average QAGS generated **1.43 Questions/** 1000 Chunk Words.
Average ratio of Words in Answer/Words in Question is **2.89.**



QAGS Results

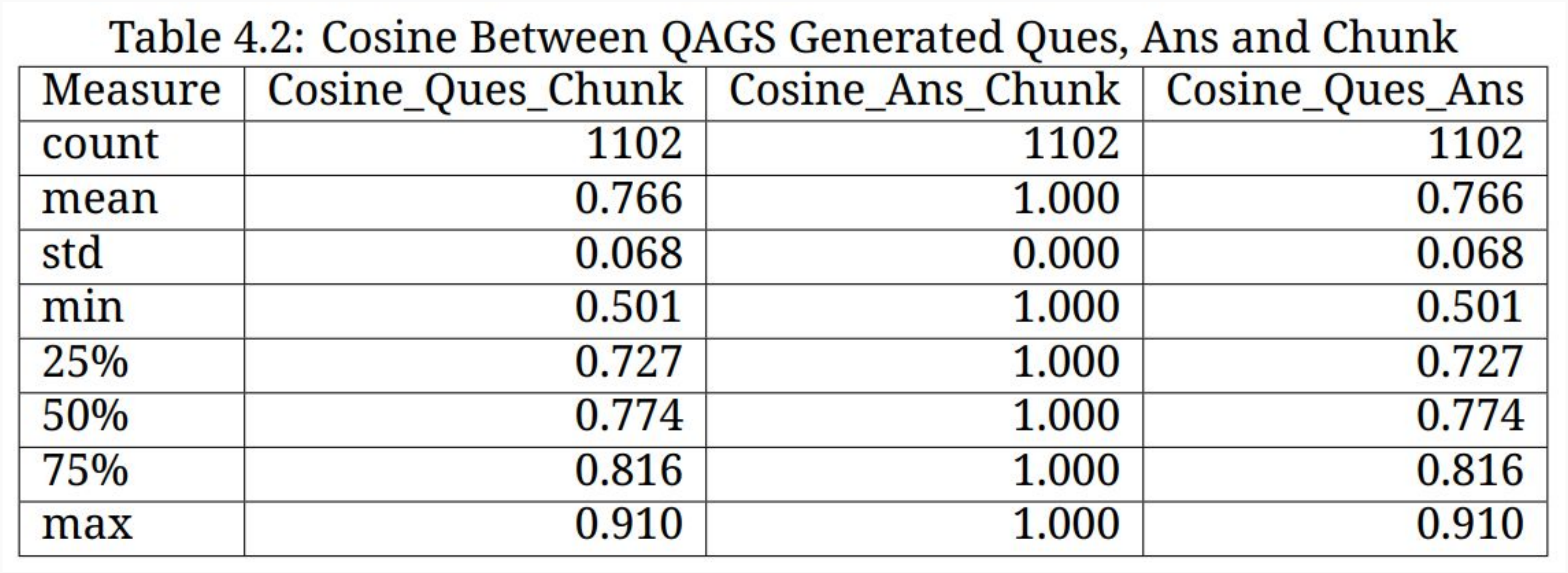
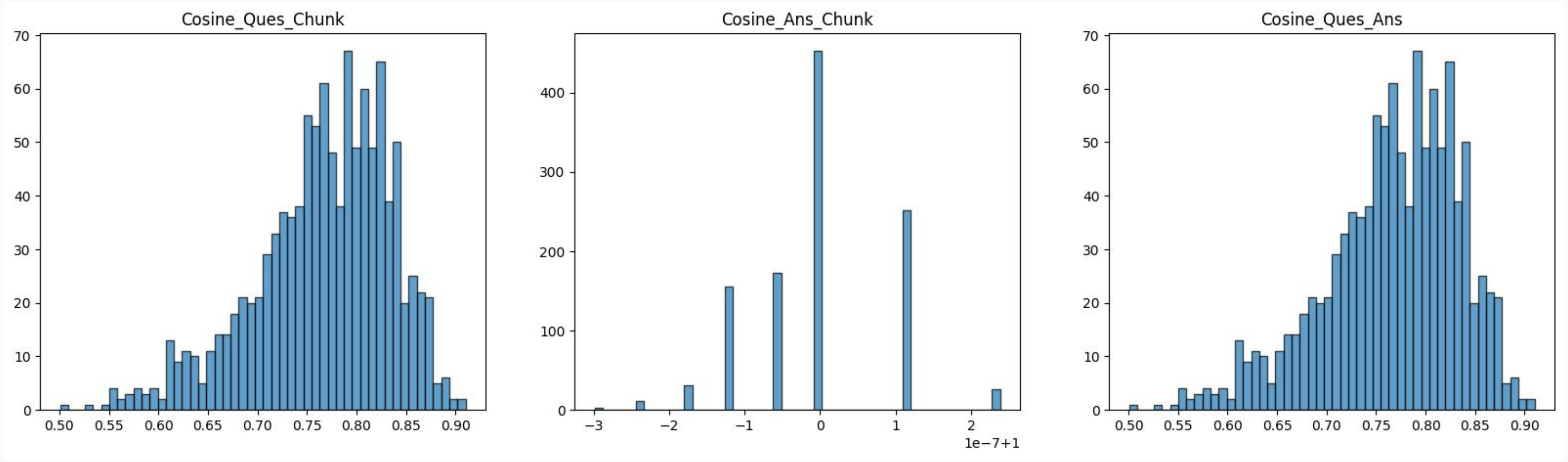


Table 4.2: Cosine Between QAGS Generated Ques, Ans and Chunk

Measure	Cosine_Ques_Chunk	Cosine_Ans_Chunk	Cosine_Ques_Ans
count	1102	1102	1102
mean	0.766	1.000	0.766
std	0.068	0.000	0.068
min	0.501	1.000	0.501
25%	0.727	1.000	0.727
50%	0.774	1.000	0.774
75%	0.816	1.000	0.816
max	0.910	1.000	0.910

Cosine Between QAGS Generated Ques, Ans and Chunk



DRS Results

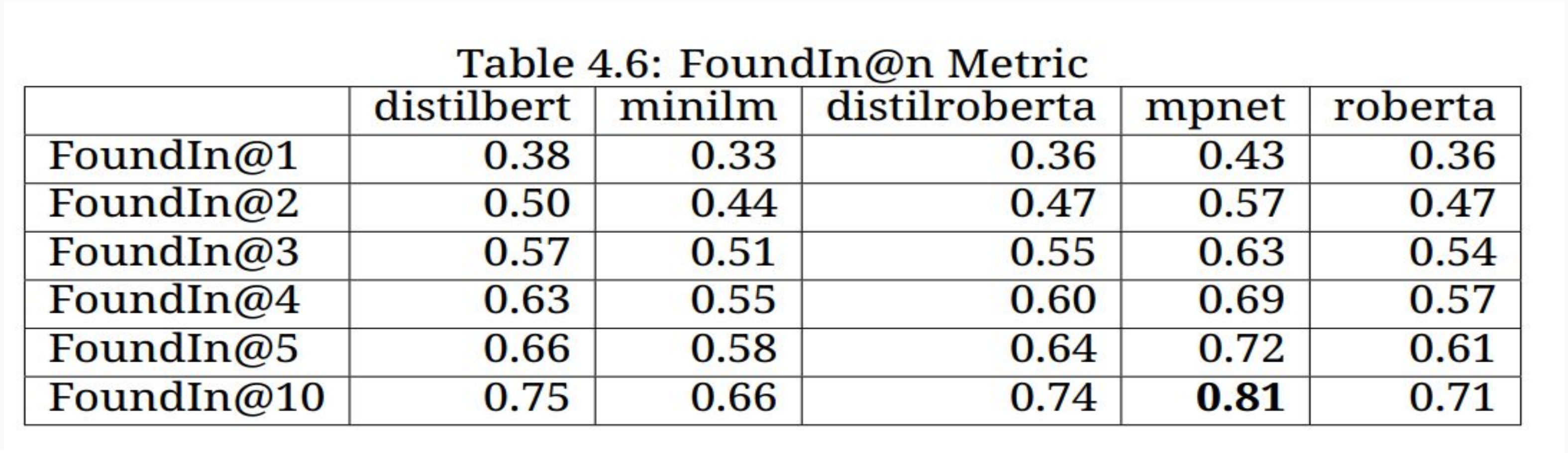


Table 4.6: FoundIn@n Metric

	distilbert	minilm	distilroberta	mpnet	roberta
FoundIn@1	0.38	0.33	0.36	0.43	0.36
FoundIn@2	0.50	0.44	0.47	0.57	0.47
FoundIn@3	0.57	0.51	0.55	0.63	0.54
FoundIn@4	0.63	0.55	0.60	0.69	0.57
FoundIn@5	0.66	0.58	0.64	0.72	0.61
FoundIn@10	0.75	0.66	0.74	**0.81**	0.71

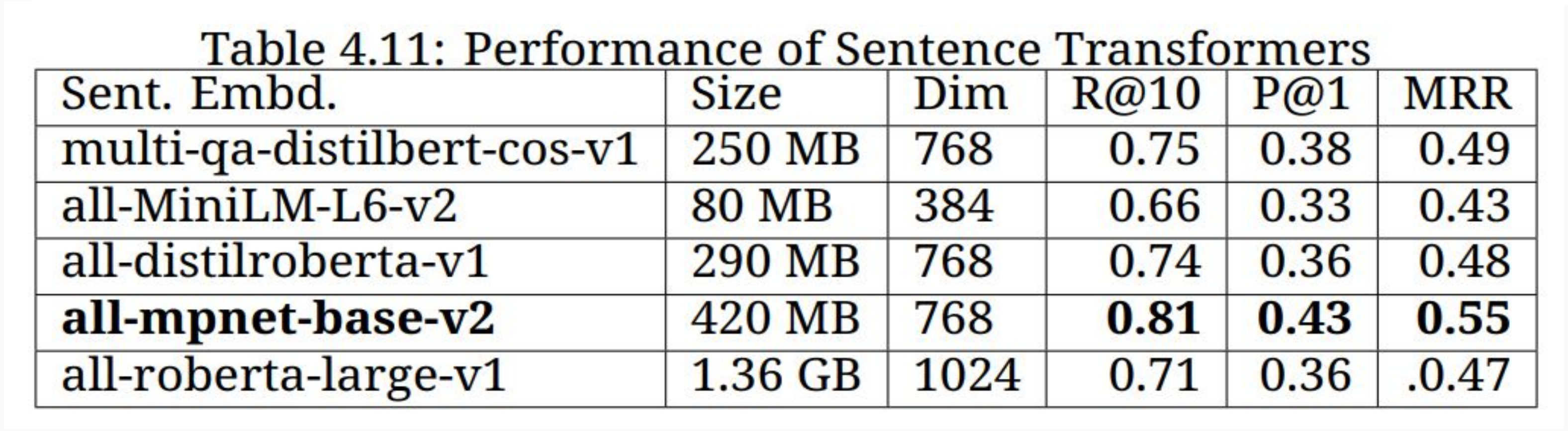


Table 4.11: Performance of Sentence Transformers

Sent. Embd.	Size	Dim	R@10	P@1	MRR
multi-qa-distilbert-cos-v1	250 MB	768	0.75	0.38	0.49
all-MiniLM-L6-v2	80 MB	384	0.66	0.33	0.43
all-distilroberta-v1	290 MB	768	0.74	0.36	0.48
all-mpnet-base-v2	420 MB	768	**0.81**	**0.43**	**0.55**
all-roberta-large-v1	1.36 GB	1024	0.71	0.36	.0.47

AGS Metrics

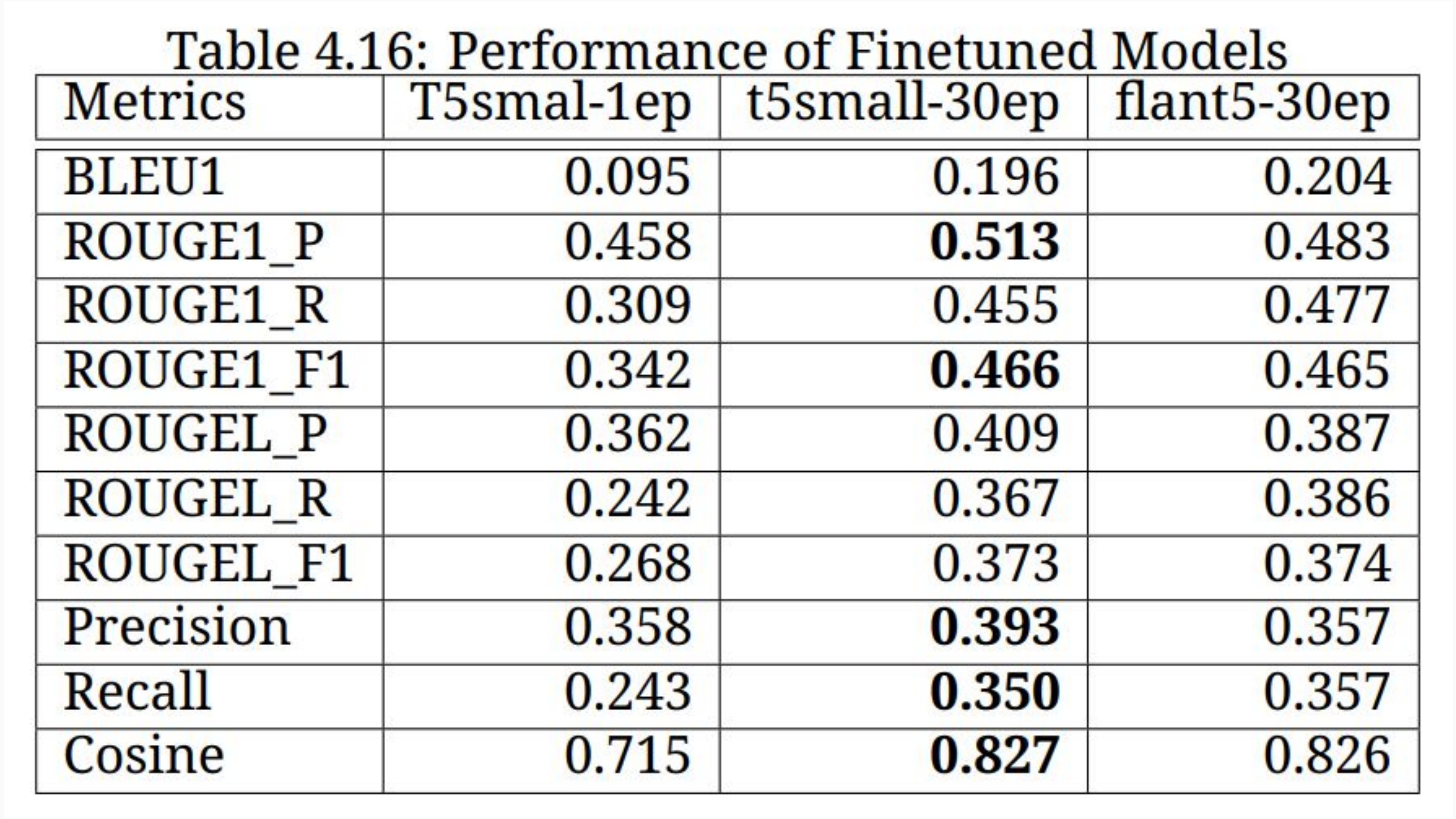
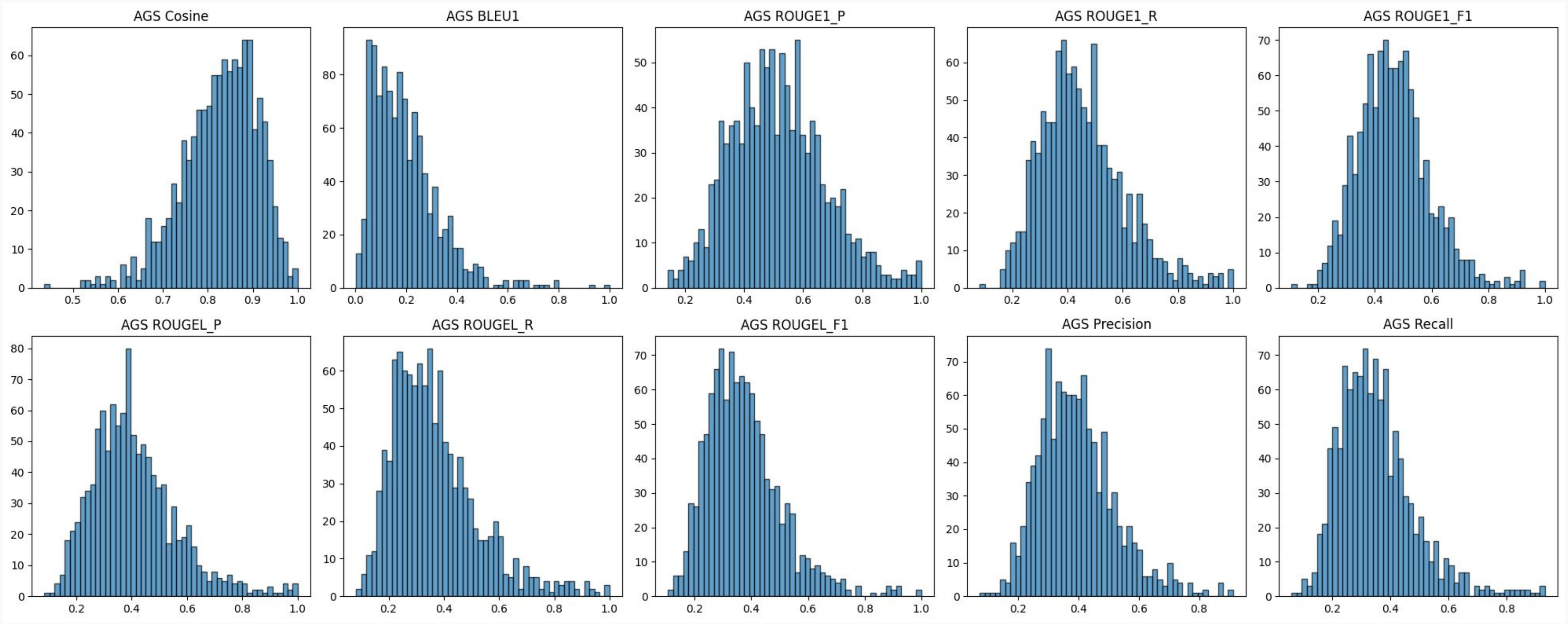


Table 4.16: Performance of Finetuned Models

Metrics	T5smal-1ep	t5small-30ep	flant5-30ep
BLEU1	0.095	0.196	0.204
ROUGE1_P	0.458	**0.513**	0.483
ROUGE1_R	0.309	0.455	0.477
ROUGE1_F1	0.342	**0.466**	0.465
ROUGEL_P	0.362	0.409	0.387
ROUGEL_R	0.242	0.367	0.386
ROUGEL_F1	0.268	0.373	0.374
Precision	0.358	**0.393**	0.357
Recall	0.243	**0.350**	0.357
Cosine	0.715	**0.827**	0.826

Distribution of Metrics on AGS Generated Answer



RAAGS Results

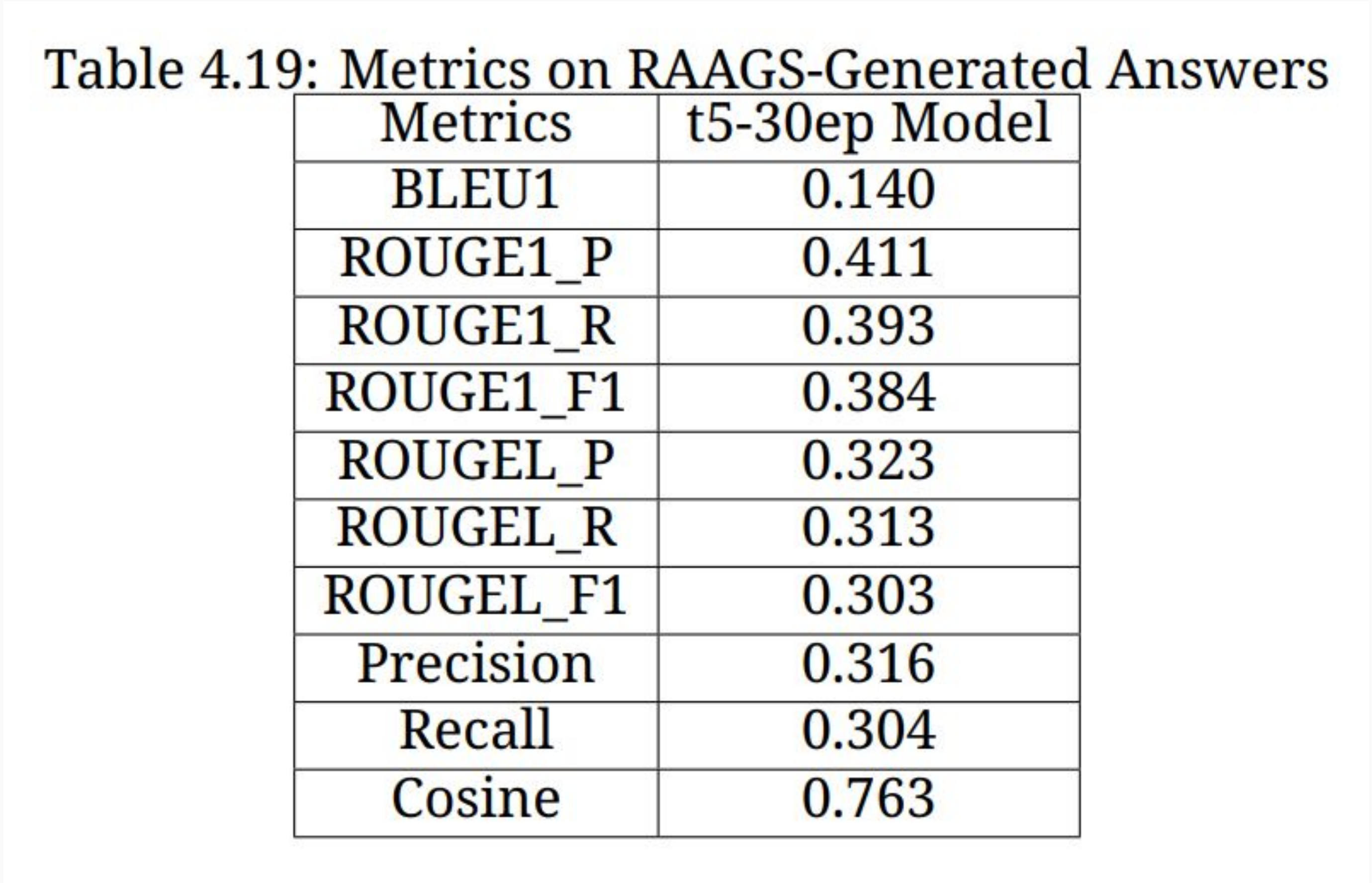
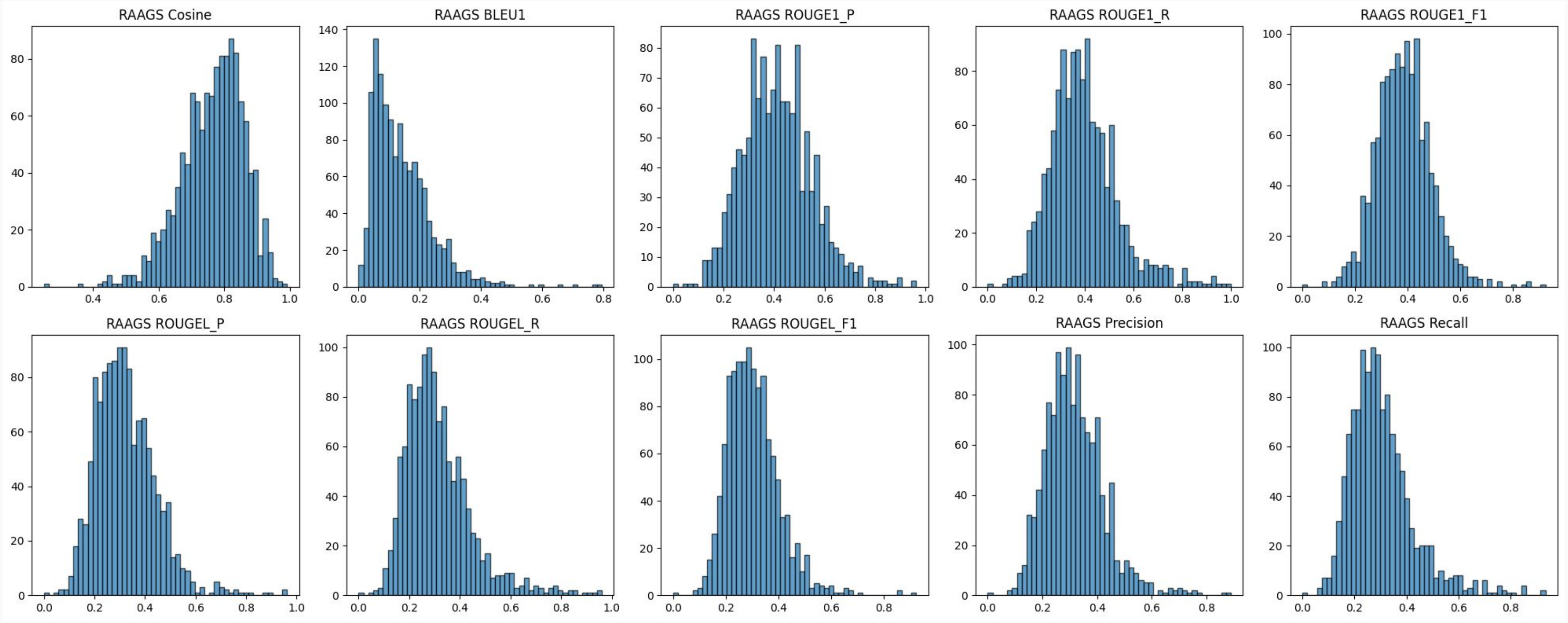


Table 4.19: Metrics on RAAGS-Generated Answers

Metrics	t5-30ep Model
BLEU1	0.140
ROUGE1_P	0.411
ROUGE1_R	0.393
ROUGE1_F1	0.384
ROUGEL_P	0.323
ROUGEL_R	0.313
ROUGEL_F1	0.303
Precision	0.316
Recall	0.304
Cosine	0.763

Distribution of Metrics on RAAGS Generated Answer



HBQAS Results (Cosine)

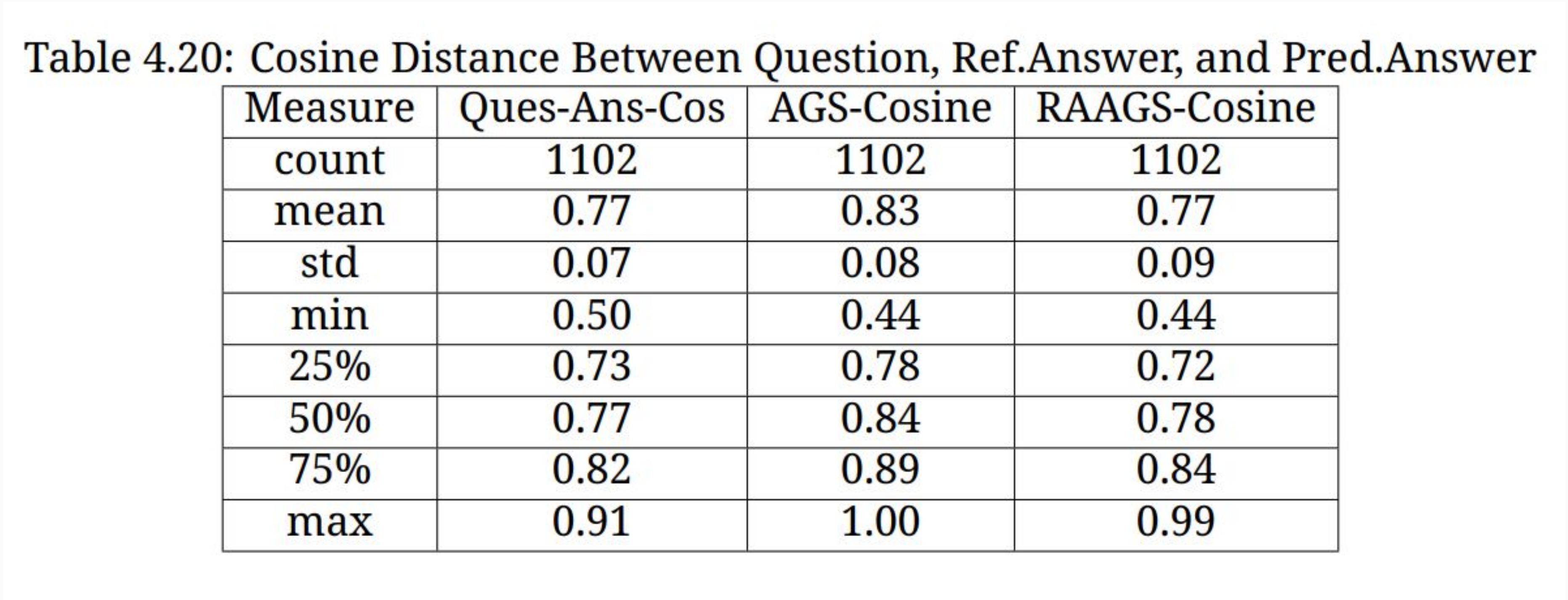
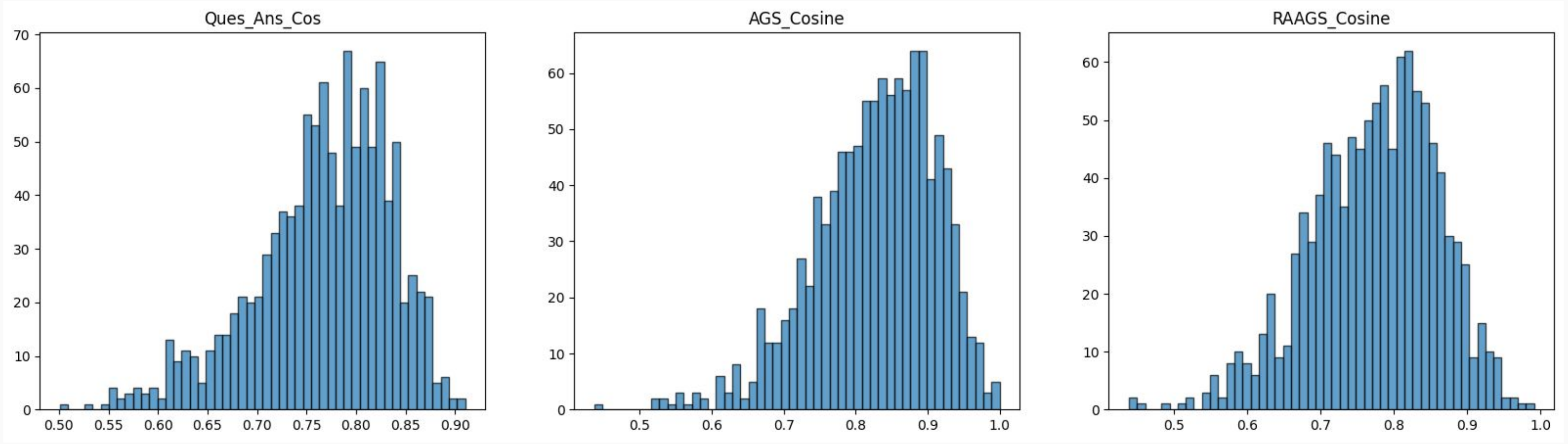


Table 4.20: Cosine Distance Between Question, Ref.Answer, and Pred.Answer

Measure	Ques-Ans-Cos	AGS-Cosine	RAAGS-Cosine
count	1102	1102	1102
mean	0.77	0.83	0.77
std	0.07	0.08	0.09
min	0.50	0.44	0.44
25%	0.73	0.78	0.72
50%	0.77	0.84	0.78
75%	0.82	0.89	0.84
max	0.91	1.00	0.99

HBQAS Cosine Between Question, Answer and Prediction



1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.

Thank you