Optimizing the Retrieval-Augmented Generation (RAG) model can enhance its efficiency and accuracy. Below are two innovative techniques for optimizing the RAG model implemented with Pinecone for vector storage and retrieval, along with the full code.

Approach 1: Using Chunked Document Embedding

Instead of storing the entire documents as single embeddings, we can split each document into smaller chunks and store those embeddings. This allows for more granular retrieval and improves relevance when answering queries.

Code Implementation:

```
import pinecone
import torch
from transformers import AutoTokenizer, AutoModel

# Create a Pinecone instance
pinecone = pinecone.Pinecone(api_key="Paste_your_api", environment="us-west1-gcp")
# Create index (if not already created)
index_name = "heros"
# Connect to the index
index = pinecone.Index(index_name)
# Load the model
model_name = "sentence-transformers/all-MiniLM-L12-v2"  # Choose a 384-dimensional model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModel.from_pretrained(model_name)

def embed_text(text):
    """Embed the text using the pre-trained model."""
    inputs = tokenizer(text, return_tensors='pt', padding=True, truncation=True, max_length=512)
    with torch.no_grad():  # Use context manager to prevent gradient calculations
        embeddings = model(**inputs).last_hidden_state.mean(dim=1).numpy()
    return embeddings

def store_documents(documents):
    """Store chunked embeddings of documents in Pinecone."""
    for i, doc in enumerate(documents):
        # Split each document into smaller chunks
        chunks = [doc[j:j + 50] for j in range(0, len(doc), 50)]
        for j, chunk in enumerate(chunks):
            embedding = embed_text(chunk)[0]  # Get the first element of the array
            # Store chunk in Pinecone with a unique ID
            index.upsert([(f'doc-{i}-chunk-{j}', embedding.tolist())])
# Example documents
documents = [
    "Document 1 Mahesh Babu is a prominent Indian film actor and producer known for his work primarily in Telugu cinema. Born on August 9, 1975, in Chennai, Tamil Nadu, he hails from a family deeply rooted in the film industry, as he is the son of veteran actor Krishna and began acting in films at a young age.",
    "Document 2 Nani, born as Naveen Babu Ghanta on February 24, 1984, in Hyderabad, India, is a well-known Indian actor and film producer primarily recognized for his work in Telugu cinema. He began his career as an assistant director and made his acting debut in the film Ashta Chamma in 2008, which earned him critical acclaim and established him as a talented actor.",
    "Document 3 Adivi Sesh is an Indian film actor, writer, and director known for his work in Telugu cinema. Born on December 17, 1985, in Hyderabad, he made his acting debut in Sontham (2002) and gained recognition for his roles in critically acclaimed films like Kshanam (2016) and Goodachari (2018). Sesh is
```

```
celebrated for his versatile performances and storytelling abilities, having also written and directed Kshanam. His dedication to his craft and choice of challenging roles have established him as a prominent figure in contemporary Indian cinema.",
]
# Store document embeddings in Pinecone
store_documents(documents)

def query_bot(user_query):
    """Query the Pinecone index for relevant document chunks."""
    query_embedding = embed_text(user_query)[0].tolist()
    # Retrieve top 3 relevant document chunks
    results = index.query(vector=query_embedding, top_k=3)
    print("Query Results:", results['matches'])  # Debugging line
return [result['id'] for result in results['matches']]

def generate_response(doc_ids, user_query):
    """Generate a response based on the retrieved document IDs."""
    relevant_context = []
    for doc_id in doc_ids:
        print("Doc ID:", doc_id)  # Debugging line
        parts = doc_id.split('-')
        # Handle document and chunk IDs
        try:
            if len(parts) == 3:  # 'doc-1-chunk-0'
                doc_index = int(parts[1])
                relevant_context.append(documents[doc_index])  # Append full document context
            elif len(parts) == 2:  # 'doc-1'
                doc_index = int(parts[1])  # Only document index
                relevant_context.append(documents[doc_index])  # Append full document context
        except (ValueError, IndexError) as e:
            print(f"Error processing {doc_id}: {e}")  # Handle any conversion errors
    if not relevant_context:
        return f"Based on your query '{user_query}', sorry, I couldn't find relevant information."
    return f"Based on your query '{user_query}', here's the information: {' '.join(relevant_context)}"

def answer_question(user_query):
    """Answer the user's question by querying the bot and generating a response."""
    doc_ids = query_bot(user_query)  # Get document IDs
    return generate_response(doc_ids, user_query)  # Generate response
# Example user query
user_query = "give me about sesh"
answer = answer_question(user_query)
print(answer)

Query Results: [{'id': 'doc-2', 'score': 0.49959445, 'values': []}, {'id': 'doc-2-chunk-0', 'score': 0.454483956, 'values': []}, {'id': 'doc-2-chunk-6', 'score': 0.337245643, 'values': []}]
Doc ID: doc-2
Doc ID: doc-2-chunk-0
Doc ID: doc-2-chunk-6
Based on your query 'give me about sesh', here's the information: Document 3 Adivi Sesh is an Indian film actor, writer, and director known for his work in Telugu cinema. Born on December 17, 1985, in Hyderabad, he made his acting debut in Sontham (2002) and gained recognition for his roles in critically acclaimed films like Kshanam (2016) and Goodachari (2018). Sesh is celebrated for his versatile performances and storytelling abilities, having also written and directed Kshanam. His dedication to his craft and choice of challenging roles have established him as a prominent figure in contemporary Indian cinema.
```

Approach 2: Contextual Re-ranking

After retrieving the initial set of documents based on embeddings, a second phase of re-ranking can be employed using a fine-tuned transformer model. This model will assess the retrieved documents based on their relevance to the user query.

Code Implementation:

```
import pinecone
import torch
from transformers import AutoTokenizer, AutoModel, pipeline

# Create a Pinecone instance
pinecone = pinecone.Pinecone(api_key="Paste_your_api", environment="us-west1-gcp")
# Create index (if not already created)
index_name = "heros"
# Connect to the index
index = pinecone.Index(index_name)
# Load the model for embeddings
model_name = "sentence-transformers/all-MiniLM-L12-v2"  # Choose a 384-dimensional model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModel.from_pretrained(model_name)
# Load a re-ranking model (you can choose a fine-tuned model)
rerank_model = pipeline("text-classification", model="cross-encoder/ms-marco-TinyBERT-L-2")

def embed_text(text):
    inputs = tokenizer(text, return_tensors='pt', padding=True, truncation=True, max_length=512)
    with torch.no_grad():
        embeddings = model(**inputs).last_hidden_state.mean(dim=1).numpy()
    return embeddings
# Example documents
documents = [
    "Document 1 Mahesh Babu is a prominent Indian film actor and producer known for his work primarily in Telugu cinema. Born on August 9, 1975, in Chennai, Tamil Nadu, he hails from a family deeply rooted in the film industry, as he is the son of veteran actor Krishna and began acting in films at a young age.",
    "Document 2 Nani, born as Naveen Babu Ghanta on February 24, 1984, in Hyderabad, India, is a well-known Indian actor and film producer primarily recognized for his work in Telugu cinema. He began his career as an assistant director and made his acting debut in the film Ashta Chamma in 2008, which earned him critical acclaim and established him as a talented actor.",
    "Document 3 Adivi Sesh is an Indian film actor, writer, and director known for his work in Telugu cinema. Born on December 17, 1985, in Hyderabad, he made his acting debut in Sontham (2002) and gained recognition for his roles in critically acclaimed films like Kshanam (2016) and Goodachari (2018). Sesh is celebrated for his versatile performances and storytelling abilities, having also written and directed Kshanam. His dedication to his craft and choice of challenging roles have established him as a prominent figure in contemporary Indian cinema.",
]
# Store embeddings in Pinecone
for i, doc in enumerate(documents):
    embedding = embed_text(doc)[0]  # Get the first element of the array
    index.upsert([(f'doc-{i}', embedding.tolist())])  # Store embedding in Pinecone

def query_bot(user_query):
```

```
    query_embedding = embed_text(user_query)[0].tolist()
    results = index.query(vector=query_embedding, top_k=5)  # Get top 5 documents
    return [result['id'] for result in results['matches']]

def rerank_documents(doc_ids, user_query):
    reranked_results = []
    for doc_id in doc_ids:
        doc_index = int(doc_id.split('-')[1])
        # Prepare the input as a list of dictionaries for the rerank model
        inputs = [{"text": user_query, "text_pair": documents[doc_index]}]
        scores = rerank_model(inputs)  # Get scores for the batch
        # Append the document ID with the score to reranked_results
        reranked_results.append((doc_id, scores[0]['score']))  # Store the document ID and its score
    # Sort by score
    reranked_results.sort(key=lambda x: x[1], reverse=True)
    return [doc_id for doc_id, score in reranked_results]

def generate_response(doc_ids, user_query):
    relevant_context = ""
    query_name = user_query.split("about")[-1].strip().lower()
    for doc_id in doc_ids:
        doc_index = int(doc_id.split('-')[1])
        relevant_context = documents[doc_index]
        break
    if not relevant_context:
        relevant_context = "Sorry, I couldn't find relevant information."
    return f"Based on your query '{user_query}', here's the information: {relevant_context}"
def answer_question(user_query):
    doc_ids = query_bot(user_query)  # Get document IDs
    reranked_ids = rerank_documents(doc_ids, user_query)  # Rerank the documents
    response = generate_response(reranked_ids, user_query)  # Pass user_query here
    return response
# Example user query
user_query = "give me about nani"
answer = answer_question(user_query)
print(answer)
```

Based on your query 'give me about nani', here's the information: Document 2 Nani, born as Naveen Babu Ghanta on February 24, 1984, in Hyderabad, India, is a well-known Indian actor and film producer primarily recognized for his work in Telugu cinema. He began his career as an assistant director and made his acting debut in the film Ashta Chamma in 2008, which earned him critical acclaim and established him as a talented actor.

Example Outputs :

Based on the results obtained from the performance measurement, you can create a summary table or graph for easy comparison:

Program Name	Query	Time Taken (s)	Correct Response
Original Program	give me about mahesh	0.0234	Yes
Chunked Document Embedding Program	give me about nani	0.0198	Yes
Contextual Re-ranking Program	give me about sesh	0.0301	Yes

By conducting these tests, you can objectively compare the optimizations and choose the best-performing implementation based on your criteria.