# 3. NVIDIA Riva: 질의응답(Question Answering, QA) 시스템 구현

## 프로젝트 개요

NVIDIA Riva Speech AI의 Dialog Manager 서비스를 활용하여, 사용자의 자연어 질문에 대해 주어진 지식 문맥(Context)에서 직접 정답을 추출하는 **Extractive QA** 시스템을 구현했습니다. (노트북: assessment for Building Conversational AI Applications.ipynb 기반)

## ⚙️ 사용 기술 및 역할

* **주요 기술:** NVIDIA Riva Speech AI (DialogManager Service), BERT Extractive QA Model
* **핵심 역할:**
  1. **QA 모델 연동:** Riva Dialog Manager를 통해 BERT 기반 QA 모델에 접근하여 질문과 문맥을 전달하는 인터페이스 구축.
  2. **Extractive QA 논리:** 모델이 문맥(knowledge\_context) 내에서 정답이 위치한 \*\*구간(span)\*\*을 정확히 예측하도록 구현하고, 높은 신뢰도(confidence)의 답변을 최종 선택하는 로직 확인.

## 💡 핵심 구현 내용

1. **Context (지식 문맥) 정의:** 질문에 대한 지식 기반이 되는 긴 텍스트 문서를 사전 정의하여 Dialog Manager의 입력으로 활용.
2. **질문-답변 예측 논리:** 질의(query)와 문맥(context)을 DialogManagerRequest로 패키징하여 Riva 서비스에 전송하고, 반환된 응답에서 최적의 답변 스팬을 추출.
3. **검증:** 모델이 문맥 내 질문에 대해서는 정확한 스팬을 추출하고, 문맥 밖의 질문에 대해서는 낮은 신뢰도 점수 또는 '답변 없음' 응답을 반환하는 것을 확인하여 시스템의 정확성을 입증.

## 📌 주요 코드 로직 (DM Prediction 함수 추상화)

import riva.client  
from riva.client.proto.riva\_nlp\_pb2 import DialogManagerRequest, Context  
  
# DM 클라이언트 초기화 (auth, service 설정)  
# ...  
  
def dm\_predict(query\_text: str, knowledge\_context: str):  
 # Context 객체 생성 (QA 모델이 검색할 지식 문맥)  
 context\_proto = Context(text=knowledge\_context)   
  
 # Dialog Manager 요청 객체 생성  
 request = DialogManagerRequest(  
 query=query\_text,   
 context=[context\_proto],  
 # 기타 파라미터: max\_answers, max\_text\_length 등  
 )  
   
 # DM 서비스 호출  
 response = riva\_dm.query(request)  
   
 # 정답 추출: 가장 높은 신뢰도의 답변 추출  
 if response.results:  
 best\_answer = response.results[0].answer  
 confidence = response.results[0].confidence  
 else:  
 best\_answer = "No answer found in context."  
 confidence = 0.0  
  
 return response