Cheat Sheet: Foundations of Multimodal AI

Concept	Description	Implementation Example

Image captioning	Generate descriptive captions for images using multimodal models.	def generate_image_caption(model, encoded_image):
		    """Generate a descriptive caption for an image."""
		    prompt = "Please provide a detailed description of this image."
		    return send_multimodal_query(model, encoded_image, prompt)
		# Example usage
		caption = generate_image_caption(model, encoded_image)
		print("Image Caption:", caption)

Image processing	Basic image processing and encoding for multimodal applications.	import base64
		from PIL import Image
		from io import BytesIO
		def encode_image(image_path):
		    """Convert image to base64 for model input."""
		    with open(image_path, "rb") as image_file:
		        encoded_string = base64.b64encode(image_file.read()).decode('utf-8')
		    return encoded_string
		def process_image(image_path, target_size=(224, 224)):
		    """Process image for model input."""
		    image = Image.open(image_path)
		    image = image.resize(target_size)
		    return image
		# Example usage
		image_path = "example.jpg"
		encoded_image = encode_image(image_path)
		processed_image = process_image(image_path)

Multimodal model setup	Basic setup for working with multimodal AI models using IBM watsonx.ai platform.	from ibm_watsonx_ai import Credentials
		from ibm_watsonx_ai.foundation_models import ModelInference
		from ibm_watsonx_ai.foundation_models.schema import TextChatParameters
		credentials = Credentials(
		    url="https://us-south.ml.cloud.ibm.com",
		)
		params = TextChatParameters(
		    temperature=0.2,
		    top_p=0.5,
		    max_tokens=2000
		)
		model = ModelInference(
		    model_id="meta-llama/llama-3-2-90b-vision-instruct",
		    credentials=credentials,
		    project_id="skills-network",
		    params=params
		)

Multimodal query	Send a combined text and image query to a multimodal model.	def send_multimodal_query(model, encoded_image, prompt):
		    """Send combined text and image query to model."""
		    messages = [{
		        "role": "user",
		        "content": [
		            {
		                "type": "text",
		                "text": prompt
		            },
		            {
		                "type": "image_url",
		                "image_url": {
		                    "url": f"data:image/jpeg;base64,{encoded_image}"
		                }
		            }
		        ]
		    }]
		    response = model.chat(messages=messages)
		    return response['choices'][0]['message']['content']
		# Example usage
		response = send_multimodal_query(model, encoded_image, prompt)
		print("Model Response:", response)

Text processing	Basic text processing and prompt engineering for multimodal applications.	def create_prompt(image_description, user_query):
		    """Create a structured prompt for multimodal analysis."""
		    prompt = f"""
		    Analyze the following image and answer the question.
		    Image Description: {image_description}
		    User Query: {user_query}
		    Please provide a detailed response that:
		    1. Describes the image content
		    2. Answers the specific question
		    3. Provides relevant context
		    """
		    return prompt
		# Example usage
		image_desc = "A cat sitting on a windowsill"
		user_question = "What is the cat doing?"
		prompt = create_prompt(image_desc, user_question)

Visual question answering	Answer questions about image content using multimodal models.	def visual_question_answering(model, encoded_image, question):
		    """Answer questions about image content."""
		    prompt = f"Please answer the following question about the image: {question}"
		    return send_multimodal_query(model, encoded_image, prompt)
		# Example usage
		question = "What color is the cat in the image?"
		answer = visual_question_answering(model, encoded_image, question)
		print("Answer:", answer)

Author

Ricky Shi