FINAL REPORT

TEAM 5

MULTIMODAL CONTENT GENERATION

Bashaarat Nawaz Mohammad, Dilip Kumar Kasina, Soumiya Rao Thada,

Vaishnavi Samboji, Venkata Kiran Reddy Kotha

SAN JOSE STATE UNIVERSITY

DATA 298B: MSDA PROJECT II

Professor Simon Shim

12-05-2025

1. Introduction

1.1 Project Background and Executive Summary

Project Background

Digital content dominates both how people express themselves online and how they learn. Students and professionals have been utilizing two kinds of spaces extensively over a period of time. These two areas are social media and educational platforms. On social networks, it is required of the creators to come up with attractive posts visually where the background audio should also go with the mood, however, performing this manually takes much time and usually one has to be a designer and a musician to have the necessary expertise. In schools, colleges, or universities, learners and teachers are getting involved more and more with multimodal materials that are a combination of the text, pictures, and diagrams plus they also get to deal with the long and the dense research which is quite hard to be done fast and efficiently.

This project develops a multimodal content generator that addresses both of these domains within a single unified system. First, it implements a social media content generator that can produce image posts and matching background music. Second, it provides an education assistant that can answer multimodal science questions and summarize long technical text. Together, these components aim to support realistic, everyday use cases in content creation and study support.

Needs and Importance

Several key needs motivated this project. Users need methods which are able to automatically generate social content that is visually and aurally attractive, without the requirement of them being expert designers or musicians. Simultaneously, the need for systems that enable multimodal question answering is increasing, which means that students can ask questions that include both text and images, for example, a diagram or a figure. Moreover, there is a demand for top-notch summarization tools that are capable of dealing with research papers and technical content so that users can get brief versions of the texts without going through every line.

By integrating image generation, music generation, multimodal question answering, and summarization into a single pipeline, the proposed system reduces manual effort and makes high-quality content creation and studying more accessible. Instead of relying on separate tools for each task, users can interact with one system that coordinates all four capabilities.

Targeted Project Problem

The project focuses on two concrete problem classes: social media content generation and education content assistance.

For social media content generation, the core problem is that users struggle to consistently create eye-catching posts with matching background audio. The proposed solution is to use a Stable Diffusion model to generate social media–ready images from text prompts and to use a finetuned musicgen_small model to generate short music clips that align with the desired mood or style of the post.

For education content assistance, the problem is that students and professionals need help with two tasks: answering science questions that involve both text and images, and summarizing long research papers or technical documents. The solution is to use a finetuned Qwen Vision-Language Model (Qwen VLM) trained on ScienceQA to answer multimodal science questions, and a finetuned T5-small model trained on an arXiv-based summarization dataset to generate concise summaries of research papers or arbitrary long text.

Motivations and Goals

The primary reasons for the project are to create AI-native tools that are simple and can be used in daily life without the need of making a large amount of content, e.g. social content creation and science learning, to delve into the doable integration of multimodal generative models (image, audio, and text) into one product, and to exemplify the way a fine-tuning of the current foundation models can be integrated in a system which is usable rather than different experiments that are isolated.

The core goals are to deliver a working system that can generate image and music content for social media from user prompts, answer science questions using both text and images, and summarize research papers or long text in a user-controllable way (for example, producing short or medium-length summaries). The project aims to demonstrate that these tasks can be achieved using Stable Diffusion (pretrained) for image generation, a finetuned musicgen_small model for music generation, a finetuned Qwen VLM on ScienceQA for multimodal question answering, and a finetuned T5-small model on arXiv summarization data for text summarization.

Planned Project Approaches and Methods

The project uses existing open-source foundation models and adapts them through fine-tuning. Stable Diffusion is used for image generation, mapping prompts to images. MusicGen-small is used for music generation, mapping prompts to audio and further improved through fine-tuning on domain-relevant data. Qwen VLM is employed for multimodal question answering, taking images and text as input and producing answers, and is finetuned on the ScienceQA dataset. T5-small is used for summarization, mapping long text to shorter text, and is finetuned on an arXiv-based summarization dataset.

A modular architecture is designed in which a controller routes user requests to the correct model. Under the hood, each model is encapsulated behind a neat, API-like interface to ensure that the parts are still decoupled and thus, easier to maintain. The results of each model are subsequently shown in a straightforward, consolidated user interface, which makes it possible for users to perform social media content creation, multimodal QA, and summarization from the identical front end.

Expected Contributions and Applications

The expected contributions of this project include a working multimodal content generation system that spans social media image and music generation as well as educational multimodal question answering and summarization. It also aims to provide a case study of how multiple fine-tuned models can be integrated into a single product pipeline. In addition, the project will produce documentation on data preparation and fine-tuning of Qwen VLM on ScienceQA and on data preparation and fine-tuning of T5-small on arXiv summarization data.

Potential applications include support for content creators and small businesses that need quick social media posts with matching background audio, as well as support for students and educators who require quick explanations of science questions that involve diagrams and condensed summaries of long research papers.

1.2 Project Requirements

The system is designed to support four primary functional use cases: social media image generation, social media music generation, multimodal question answering for education, and text summarization. Each use case is backed by a specific model and a clear input–output flow, as described below.

Functional Requirements

For Social Media Image Generation, the system should allow the user to enter a natural language prompt describing a social media post (for example, "a minimal aesthetic desk setup with a laptop and coffee for a productivity reel thumbnail"). This prompt is passed to a Stable Diffusion model, which generates one or more candidate images. The user must then be able to view the outputs in the interface, download them, or reuse them for further content creation.

For Social Media Music Generation, the user specifies the desired musical style or mood (for example, "lofi chill", "upbeat electronic", or "cinematic") and can optionally specify the desired length of the clip. The system employs a finetuned musicgen_small model to create a sound file that corresponds to the specified conditions. The user can listen to the created clip in the interface and save it on their device if they desire to use it elsewhere.

For Multimodal Question Answering (Education), the user inputs a science question in the form of text, which can be the question statement, answer options, and/or any other context. The user is also allowed to upload an image like a diagram or figure that goes with the question. The system passes both the text and the image inputs to the finetuned Qwen VLM. The model returns a text answer to the question, and, if configured, may also provide an explanation of the reasoning or the correct option.

For Text Summarization, the user provides a block of text, which can be a research paper abstract, an introduction, or a full section of a technical document. This text is passed to the finetuned T5-small model. The system returns a summarized version of the input and, if there are any implemented controls, the user is able to select the length or level of detail (e.g. short versus medium summaries).

Besides the core functional requirements, the system should also exhibit several non-functional behaviors:

- Provide reasonable latency across all four tasks.
- Support GPU execution for both inference and fine-tuning.
- Return clear, informative error messages if inputs are missing, malformed, or invalid.

AI-Powered Feature Requirements

Each model-powered feature must satisfy a specific set of behaviors:

Stable Diffusion

- Accept natural language prompts from users.
- Generate images suitable for social platforms, with resolution and output format configurable.

musicgen_small

- Generate audio that broadly matches the requested mood or style.
- Produce audio of a fixed target duration (for example, N seconds) if the system is configured to enforce a specific length.

Qwen VLM

- Accept combined text and image inputs as a single multimodal query.
- Produce coherent, scientifically relevant answers for ScienceQA-style questions.

T5-small

- Generate shorter summaries that preserve the key ideas of arXiv-like input text.

Measurable Requirements (High-Level)

Where possible, model performance and output quality should be evaluated using quantitative or structured qualitative metrics:

- For the finetuned Qwen VLM on ScienceQA, performance is measured using accuracy or exact match on an evaluation subset of the dataset.
- For T5-based summarization, quality can be measured with standard automatic metrics such as ROUGE or BLEU, and/or with human judgment of informativeness and faithfulness.
- For social media content (images and music), evaluation is more qualitative; user-perceived relevance to the prompt, aesthetic quality, and overall usefulness for social media posts can be collected via user studies or informal feedback.

1.2.1 Formal Functional Requirements

Table 1 summarizes the core functional requirements for the multimodal content generator.

ID	Requirement Name	Description	Type	Metric / Evidence	Target (to be finalized)	Verification Method
FR-01	Social Image Generation	The system shall generate at least one social media–ready image from a user text prompt.	Functional	Successful image generation rate	≥ 95% of valid prompts produce an image	Functional test; manual inspection
FR-02	Multi-Image Suggestions (Optional)	The system should support generating multiple candidate images per prompt.	Functional	Number of images returned per request	1–N, configurable	Functional test

FR-03	Social Music Generation	The system shall generate a short music clip from a user's style/mood description.	Functional	Successful audio generation rate	≥ 95% of valid prompts produce an audio clip	Functional test; manual listening check
FR-04	Audio Playback and Download	The system shall allow users to play and download generated music clips.	Functional	Playback and download success rate	≥ 99%	UI test; file integrity check
FR-05	Multimodal QA Input Handling	The system shall accept a science question as text and optionally	Functional	Ability to upload/select image + text	All required fields usable	UI test; integration test

		an associated image.				
FR-06	Multimodal Science QA Answering	The system shall return a text answer for a given science question (text ± image) using Qwen VLM.	Functional	Answer produced per valid query	≥ 99% of valid queries return an answer	Functional test; log inspection
FR-07	ScienceQA-Style Accuracy	The fine-tuned Qwen VLM shall achieve acceptable accuracy on a ScienceQA-style evaluation subset.	Functional	Accuracy / exact match on evaluation set	≥92%	Offline evaluation on held-out ScienceQA

FR-08	Text Summarization	The system shall generate a shorter summary for a user-provided text using the finetuned T5-small.	Functional	Summary generated per valid text input	≥ 99% of valid inputs return a summary	Functional test
FR-09	Scientific Summary Quality	Summaries of research-style text shall preserve key ideas at a reduced length.	Functional	ROUGE / human relevance scores	≥ target ROUGE / human rating threshold	Offline ROUGE evaluation; small user study
FR-10	Mode Selection	The system shall allow users to switch between Social Media and Education	Functional	Mode selection working as intended	100%	UI test

		modes from the UI.				
FR -11	Error Handling	The system shall present clear error messages for missing/invalid inputs.	Functional	Presence and clarity of error messages	All invalid cases produce informative message	Negative test cases; UI review

1.2.2 Non-Functional Requirements

Table 2 summarizes key non-functional requirements. Values are expressed as design targets and can be refined based on hardware and deployment constraints.

ID	Requirement Category	Description	Metric / Indicator	Target (to be finalized)	Verification Method

NFR-01	Performance – Latency (Image)	Average end-to-end response time for Stable Diffusion image generation.	Mean latency per request	≤ Y_1 seconds under nominal load	Load/latency testing
NFR-02	Performance – Latency (Music)	Average end-to-end response time for musicgen_small music generation.	Mean latency per request	≤ Y_2 seconds for target clip length	Load/latency testing
NFR-03	Performance – Latency (QA)	Response time for Qwen VLM multimodal QA.	Mean latency per query	≤ Y_3 seconds	Load/latency testing
NFR-04	Performance – Latency (Summ.)	Response time for T5-small summarization.	Mean latency per summarization request	≤ Y_4 seconds	Load/latency testing

NFR-05	Availability	System availability during scheduled demo/usage window.	Uptime percentage	≥ 99% within evaluation period	Monitoring; manual logging
NFR-06	Robustness	System behavior under invalid or extreme inputs (empty text, oversized files, etc.).	Graceful degradation; no crashes	All invalid inputs handled without crash	Negative tests; fault injection
NFR-07	Usability	Ease of use of the UI for non-technical users.	SUS score or equivalent usability score	≥ target SUS (e.g., ≥ 70,)	Small user survey; heuristic evaluation
NFR-08	Maintainability	Ease of modifying or swapping	Modularity of codebase;	Models encapsulated in	Code inspection

		underlying models.	separation layers	independent services	
NFR-09	Portability	Ability to run the system on different GPU machines with minimal changes.	Number of environment-specific changes	Minimal environment-specific configuration	Deployment on ≥ 2 different machines
NFR-10	Security	Protection against arbitrary code execution via user inputs; limited file handling.	Input validation, restricted file types	All upload and prompt interfaces validated	Code review; basic penetration testing

1.3 Project Deliverables

The main deliverables of the multimodal content generator project are:

Multimodal Content Generator System

Integrated application providing:

- Social media image generation using a Stable Diffusion model.
- Social media music generation using a finetuned musicgen_small model.

- Multimodal science question answering using a finetuned Qwen VLM model trained on ScienceQA.
- Text summarization using a finetuned T5-small model trained on an arXiv summarization dataset.

Model Artifacts

- Configuration and reference information for the Stable Diffusion model used for social media image generation.
- Finetuned checkpoint(s) for musicgen_small used in the system.
- Finetuned checkpoint(s) for Qwen VLM on the ScienceQA dataset.
- Finetuned checkpoint(s) for T5-small on the arXiv summarization dataset.

Source Code Repository

- Backend code implementing model invocation, pre-/post-processing, and orchestration logic.
- Frontend/UI code providing user interaction for both the Social Media and Education modes.
- Utility scripts for data preprocessing, training, and evaluation of the finetuned models.

Project Documentation

- The final technical report (this document) detailing data engineering, model development, system design, evaluation, and conclusions.
- Internal documentation or README files describing the environment setup, execution instructions, and usage guidelines.

Presentation Materials

- Final project presentation slides that provide a brief overview of the objectives, methodology, system architecture, results, and future work.

Demonstration Artefacts

- Live or recorded demonstration of the system, illustrating:
    - End-to-end social media content generation (image + music).
    - Multimodal science QA examples.
    - Summarization of research-style text.
- Example inputs and outputs (images, audio clips, QA exchanges, and summaries) used during demonstration.

1.4 Technology and Solution Survey

This section briefly surveys alternative technologies for each core function—image generation, music generation, multimodal question answering, and text summarization—and explains the rationale for selecting the technologies used in the project.

1.4.1 Image Generation Technologies

Generative image models initially were based on GANs, like StyleGAN and BigGAN, which can create images of high quality but are generally hard to train and not very adaptable for text-conditioned generation. After that, diffusion models are now widely used for text-to-image synthesis. Latent Diffusion Models (LDMs), popularized as Stable Diffusion, operate in a compressed latent space and provide high-resolution image generation with significantly reduced computational cost compared to pixel-space diffusion.

Alternative proprietary or closed systems such as DALL·E and Midjourney offer strong image quality but are not open-source and cannot be finetuned or integrated as flexibly into research projects.

Rationale for Stable Diffusion in this project:

- It has a well-supported ecosystem, is open-source, and has been widely adopted.
- Because of latent-space diffusion, it offers a good trade-off between quality and computational efficiency.
- Easy integration with existing Python and deep learning toolchains, enabling fast experimentation with prompts and downstream social media use cases.

1.4.2 Music Generation Technologies

Traditional symbolic music models (e.g., RNN-based sequence models over MIDI) capture musical structure but often struggle with audio-level realism. Recent work has moved towards audio-native generative models, including Jukebox and AudioLM-style approaches, but many are large and difficult to deploy.

MusicGen, introduced as part of Meta's AudioCraft toolkit, is a transformer-based model that generates audio tokens via an EnCodec tokenizer and is specifically designed for text-conditioned music generation. It is released in multiple sizes (including musicgen_small), making it suitable for resource-constrained environments.

Rationale for MusicGen-small in this project:

- Openly available text-to-music model with pre-trained weights and code.
- Smaller variant (musicgen_small) is more feasible to finetune and deploy within academic GPU limits.
- Natively supports text prompts describing style or mood, which aligns well with social media content generation.

1.4.3 Multimodal Question Answering Technologies

Multimodal QA requires joint reasoning over text and images. Early approaches combined CNN-based visual encoders with separate text encoders for visual question answering (VQA). More recent Vision-Language Models (VLMs) and Large Vision-Language Models (LVLMs) unify image and text processing within transformer-based architectures.

The ScienceQA benchmark introduces ~21k multimodal multiple-choice science questions with rich annotations, including lectures and explanations, and is specifically designed to evaluate multimodal reasoning in science education settings.

The Qwen-VL family extends the Qwen language model with visual perception, including a visual receptor and multimodal training pipeline, achieving strong performance on diverse vision-language benchmarks.

Alternative multimodal models include Flamingo, BLIP-2, LLaVA, and other LVLMs, but these may be less tailored to ScienceQA-style tasks or have more restrictive licenses or model access.

Rationale for Qwen VLM in this project:

- Offers a unified architecture for text and image inputs with strong reported performance on visual understanding and QA benchmarks.
- Open-source availability and tools for finetuning facilitate adaptation to the ScienceQA dataset.
- Directly supports the multimodal input format used in the project (image + text → answer).

1.4.4 Text Summarization Technologies

Abstractive summarization has changed sequentially from RNN-based encoder–decoder models to transformer-based models like BART, PEGASUS, and T5. Among those, the T5 (Text-to-Text Transfer Transformer) model is the one that reformulates every language task, summarization included, as a text-to-text problem and achieves excellent transfer learning results on various benchmarks.

For long scientific documents like arXiv papers, models must handle technical vocabulary and longer context windows. T5-small is a compact member of the T5 family, allowing finetuning within typical academic compute budgets while still benefiting from the text-to-text formulation.

Rationale for T5-small in this project:

- Well-established text-to-text architecture with existing tooling and examples for summarization tasks.
- Smaller model size eases fine tuning on arXiv summarization data in constrained environments.

- Naturally fits the “long text → short summary” requirement without changing the model interface.

1.5 Literature Survey of Existing Research

This subsection summarizes key lines of prior work relevant to the multimodal content generator: social media content generation, music generation, multimodal science QA, and scientific text summarization.

1.5.1 Generative Image Models for Content Creation

Rombach et al. presented Latent Diffusion Models (LDMs) that carry out diffusion in a learned latent space instead of pixel space, thus being able to create high-resolution images at a fraction of the computational cost. Stable Diffusion, a popular text-to-image generation method, is basically an extension of this research. Follow-up research has explored improved control and editing in diffusion models, including semantic steering in latent space.

These models have become standard tools for content creation workflows, enabling non-experts to generate customized images from textual prompts, which directly motivates the social media image generation component of this project.

1.5.2 Neural Music Generation

Recent work on music generation has shifted from symbolic representations to audio-native generative models. Meta’s MusicGen is a transformer-based model trained on licensed music, using EnCodec for neural audio tokenization and generating high-fidelity audio conditioned on text descriptions or melodies. The AudioCraft framework open-sources MusicGen, AudioGen, and EnCodec, enabling research and practical applications in text-to-music synthesis.

These developments demonstrate that controllable, text-driven music generation is feasible and suitable for applications such as video background tracks and social media content—exactly the scenario addressed by the music component of this project.

1.5.3 Multimodal Science Question Answering and Vision-Language Models

The ScienceQA dataset was introduced to study multimodal science question answering with rich annotations, including lectures and explanations that support chain-of-thought reasoning. ScienceQA contains over 21k multiple-choice questions spanning natural science, language science, and social science, many of which include both text and images. This dataset has become a standard benchmark for evaluating multimodal reasoning and explanation capabilities of AI systems.

In parallel, the Qwen-VL series extends the Qwen language model to the vision-language setting via a visual receptor and a multimodal training pipeline, achieving strong performance on a variety of LVLM benchmarks. This line of work showcases how large-scale pretraining and multimodal fusion can support tasks requiring joint understanding of images and text.

The educational QA module in this project is positioned at the intersection of these works, combining a Qwen VLM model with the ScienceQA dataset to deliver multimodal science question answering.

1.5.4 Scientific Text Summarization and T5

Raffel et al. proposed the T5 (Text-to-Text Transfer Transformer) framework, showing that casting diverse NLP tasks into a unified text-to-text format enables effective transfer learning across many benchmarks. T5 has been widely applied to summarization tasks, with various model sizes (including T5-small) supporting finetuning under different resource constraints.

Later work has primarily been centered on condensing lengthy and complicated documents, which also involves scientific papers, by employing not only T5 but also various other transformer-based architectures. These approaches usually perform fine-tuning on a domain-specific corpora like arXiv, thus allowing models to become more proficient in dealing with technical jargon and the organization of lengthy documents.

The summarization component of this project builds directly on this literature by finetuning T5-small on an arXiv-based summarization dataset to generate concise summaries of research papers and related content.

1.5.5 Positioning of This Project

Unlike previous work, this project is not about a new model architecture, but rather:

- It takes state-of-the-art or most commonly used models (Stable Diffusion, MusicGen, Qwen VLM, T5) and merges them into one multimodal application that covers social media and educational use cases.
- It changes Qwen VLM for the ScienceQA benchmark and T5-small for arXiv-style summarization with the definite intention of helping students and researchers.

- It is writing, imaging, and music generation plus multimodal QA and scientific summarization merged into one end-to-end system targeted at content creators and learners' real-world workflows.

2. Data and Project Management Plan

2.1 Data Management Plan

For this phase of the project, the data management plan focuses less on collecting new raw data and more on systematically managing curated machine learning datasets, derived training splits, model outputs, and experiment artifacts used for fine-tuning and evaluating the models.

From the standpoint of data collection, the project is dependent on well-known open-source datasets rather than the newly created ones by scraping. The main datasets are: (i) the ScienceQA dataset for multimodal science question answering, (ii) an arXiv-based scientific text summarization dataset for training T5-small, and (iii) a music dataset of short clips paired with style or mood descriptors for fine-tuning musicgen_small. These datasets are downloaded from reliable sources like Hugging Face and other open sources under their respective licenses.

For data handling, the datasets are each structured in a shared project framework with different subfolders for the unprocessed data, intermediate processed data, and the final train/validation/test splits. File and folder names follow a consistent convention (dataset name, split, version, and date), which makes it possible to reproduce experiments and trace any model run back to the exact data version used. HF Datasets scripts and preprocessing notebooks act as the "data contract": they define how raw files are transformed into model-ready tensors (for example, tokenized sequences for T5-small and Qwen VLM, or audio tensors for MusicGen).

Updates to preprocessing logic are tracked in Git, and any substantial change (such as adjusting maximum sequence length or filtering specific question types) is documented in commit messages and experiment notes.

Regarding storage methods, all project data—raw datasets, processed splits, model checkpoints, and logs—is stored in Google Cloud Storage (GCS) buckets, with synchronized working copies on a GPU-enabled machine for training and experimentation. Raw datasets in GCS are treated as read-only, while transformed datasets and experiment outputs are written to clearly separated paths to avoid accidental overwrites. Model checkpoints for Qwen VLM, T5-small, and musicgen_small are stored in dedicated GCS locations with explicit run identifiers (model name, dataset, date, and key hyperparameters), which simplifies mapping evaluation results back to specific training runs and supports rollback if needed. Regular backups of important artifacts (final checkpoints, key logs, and selected outputs) are maintained within GCS using bucket versioning and replicated folders.

For usage mechanisms, access to GCS buckets and local data directories is restricted to project team members through account-level permissions and controlled sharing. No personally identifiable information is involved, as all datasets are public benchmark or research corpora. Data is only such that it serves model training, validation, evaluation, and demonstration in the context of this project. Inference jobs and training scripts that consume data are done through standardized loaders which are based on HF Datasets and PyTorch, thus ensuring that the same preprocessing is done for all the experiments. This focused data management approach emphasizes reproducibility, traceability, and safe handling of research datasets rather than large-scale external data collection or sharing.

2.2 Project Development Methodology

The development work in this phase was based on an iterative, experiment-driven methodology that focused on model development, system integration, and evaluation. Instead of handling the project as a kind of linear pipeline, we mapped out the work in short cycles through which we updated one component, assessed it, and then merged it with the wider system.

The first set of iterations focused on model development. For each task—multimodal QA, summarization, and music generation—we started by standing up a minimal but functioning training pipeline using the chosen base models (Qwen VLM, T5-small, and musicgen_small) and the prepared datasets. This included setting up training scripts, defining loss functions, configuring data loaders, and establishing basic logging of metrics such as training loss and validation accuracy (for ScienceQA) or ROUGE scores (for summarization). Initially, the experiments were intentionally limited in scale, such as having less epochs or smaller data subsets, in order to confirm the correctness of the pipeline without making a longer training run.

After stabilizing the core pipelines, the work changed to systematic fine-tuning and hyperparameter adjustment. For each model, we tried different learning rates, batch sizes, and training schedules, and at the same time, we checked validation metrics and qualitative examples. These experiments were performed in small groups and recorded through experiment logs, which consisted of the setup, random seeds, and the results obtained.

When a configuration showed clear improvement over the previous baseline, it was promoted as the new default for that model.

After the individual models reached acceptable performance levels, the development focus moved to system-level integration. The methodology here was to build vertical "slices": for example, connecting the Qwen VLM model to a backend endpoint, then wiring that endpoint to a simple UI element for uploading an image and entering a question, and finally testing the end-to-end flow with real ScienceQA-style examples. A similar slice was implemented for T5 summarization, and then for the social media image and music generation paths. This vertical-slice approach ensured that integration problems (such as mismatched input formats or timeouts) were exposed early rather than at the end.

Throughout this period, Git and a shared repository were used to coordinate work. Each major change (new model configuration, new endpoint, UI feature, or evaluation script) was committed with descriptive messages, and branches were used for more risky experiments. Informal "checkpoints" were set at the end of each major milestone—such as "Qwen VLM fine-tuning stable on ScienceQA" or "all four main endpoints integrated into the backend"—to track progress and guide the next iteration.

Finally, the methodology included a dedicated evaluation and polishing stage. Once all core features were in place, we designed targeted tests for each user-facing function, collected qualitative examples, and identified failure modes. The feedback from these tests led to small refinements in both model settings (e.g., generation parameters) and UI behavior (e.g., clarifying instructions or input validation). This combination of iterative model experimentation, vertical integration, and focused evaluation formed the backbone of the development methodology for this phase.

2.3 Project Organization Plan

The project in this phase was organized around both functional streams and clearly defined phases so that work could progress incrementally while still converging to a single integrated system. We worked collaboratively across all major activities, with responsibilities shared rather than locked into rigid roles. Broadly, the tasks were divided into four functional streams: (i) model development (fine-tuning Qwen VLM, T5-small, and musicgen_small), (ii) data and experimentation support (preprocessing, splits, experiment logging), (iii) backend and orchestration (endpoints, controller, error handling), and (iv) user interface and deployment (UI flows and running prototype). These streams were continuously interacting, but they were managed within an explicit work breakdown structure (WBS) which converted the project into phases, deliverables, and work packages corresponding to the time from Aug 29 to Dec 3.

<u>Phase 1 – Model Development and Intelligent Solution Design (Aug 29 – Sep 29)</u>

This phase focused on defining and implementing the core intelligent solutions and establishing stable model pipelines. The main deliverable was D1: Fine-tuned model candidates and documented intelligent solution design.

- <u>WP1.1 Model Proposals and Selection</u>

  Confirm the mapping from project tasks to models, namely Stable Diffusion for social media image generation, musicgen_small for music generation, Qwen VLM for multimodal science QA, and T5-small for scientific text summarization, and document their roles within the overall system.

- <u>WP1.2 Model Supports and Training Pipelines</u>

  Set up training and evaluation pipelines for Qwen VLM, T5-small, and musicgen_small, including dataset loading, preprocessing, loss definitions, logging, and configuration management so that each model could be fine-tuned reproducibly.

- <u>WP1.3 Model Comparison and Baseline Evaluation</u>

  Run baseline fine-tuning experiments for the selected models, compare different configurations using validation metrics and qualitative examples, and choose the configurations to carry forward as the basis for later integration.

- <u>WP1.4 System Requirements Analysis</u>

  Refine the system boundary and actors (social media content creators, students, educators) and identify the main use cases that the system must support, such as social media post generation, multimodal science QA, and research text summarization.

- <u>WP1.5 System Design and Intelligent Solution Specification</u>

  Specify the high-level architecture, including frontend, backend controller, model services, and data/model store, and describe how the four models are orchestrated to implement the intelligent solution for the targeted problems.

- <u>WP1.6 System Supporting Environment Description</u>

  Document the technical environment that supports development and execution (GPU-enabled setup, Python, PyTorch, Hugging Face ecosystem, Google Cloud Storage) and how it is used for model training and system runs.

<u>Phase 2 – Model Execution Analysis, System Quality, and Visualization (Sep 30 – Nov 3)</u>

This phase concentrated on understanding how the models behave in practice and how the integrated system performs from a quality perspective. The main deliverable was D2: Validated model behavior and system-level analysis with supporting visualizations.

- <u>WP2.1 Analysis of Model Execution and Evaluation Results</u>

  Analyze training and validation runs for Qwen VLM, T5-small, and musicgen_small by reviewing metrics (such as validation accuracy or loss) and inspecting representative outputs to characterize typical successes and failures.

- WP2.2 System Quality Evaluation

  Evaluate the correctness and stability of the integrated model functions from an end-to-end viewpoint, focusing on whether the generated answers, summaries, images, and music clips are usable for their intended educational and social media scenarios.

- WP2.3 System Visualization Artefacts

  Produce visual and illustrative artefacts—such as selected QA examples, input text versus generated summaries, sample generated images and music clips, and plots of key metrics—that can be used in the report and presentation to communicate system behavior.

Phase 3 – Deployment, User Interface, and Final Deliverables (Nov 4 – Dec 3)

The final phase aimed to deliver a working prototype and the associated documentation and demo materials. The main deliverable was D3: Deployed prototype with integrated UI plus final report and demo materials.

- WP3.1 Deployment of Backend and Model Services

  Deploy the backend controller and model endpoints in a runnable environment, ensuring that all four model services (Stable Diffusion image generation, musicgen_small music generation, Qwen VLM multimodal QA, and T5-small summarization) can be invoked reliably from the server side.

- WP3.2 User Interface Implementation and Integration

  Implement the user interface for the Social Media and Education modes, connect UI actions to the backend endpoints, and verify that users can go from inputs (prompts, questions, images, text) to outputs (images, music, answers, summaries) in a single flow.

- WP3.3 System-Level Testing of Deployed Prototype

  Carry out main use case end-to-end tests. Confirm that the deployed system is functioning as expected, that error conditions are handled in a user-friendly way, and that the performance is good enough for a demo.

- WP3.4 Final Report Preparation

  Gather the project report into one document and polish it up. Include the descriptions of data management, model development, system design, evaluation results, and conclusions that together make up a coherent report.

- WP3.5 Presentation and Demo Preparation

  Prepare the presentation slides and demo scenario, including selecting representative prompts, questions, summaries, images, and music clips, and organizing them into a clear demonstration of the system's capabilities.

Across all three phases, work packages were carried out collaboratively by the team members, with frequent coordination to handle interdependencies—for instance, UI integration depending on stable backend endpoints, and evaluation visualizations depending on finalized model outputs. This hierarchical and incremental organization allowed the project to progress in a controlled way from model design to a complete deployed system.

2.4 Project Resource Requirements and Plan

The resource plan for this phase was shaped by the need to fine-tune three different models and run repeated inference for evaluation and demonstration. The primary hardware requirement was access to a GPU-equipped environment capable of handling moderately sized vision-language and sequence-to-sequence models. In practice, this meant using a machine with a recent NVIDIA

GPU (with sufficient VRAM to load Qwen VLM and Stable Diffusion), a multi-core CPU, and enough RAM to support data loading and preprocessing without bottlenecks. Fast SSD storage was crucial for model checkpoints and dataset caching, especially when using Hugging Face Datasets and Diffusers.

On the software side, the main stack was Python, PyTorch, and the Hugging Face ecosystem (Transformers, Diffusers, and Datasets). These libraries supplied most of the functionality needed for loading the base models, defining fine-tuning procedures, and implementing efficient data pipelines. In addition, utilities like NumPy, pandas, and standard Python logging were employed for data manipulation and experiment tracking. For the backend, a lightweight web framework (such as FastAPI or Flask) was planned and used to expose model endpoints, while the frontend was done using standard web technologies that are compatible with the course constraints.

Cloud services were considered an optional extension rather than a core dependency. The plan allowed for cloud GPUs to be used for longer training runs if local resources became a bottleneck, but most experiments were supposed to be conducted on a shared local GPU environment in order to keep the costs low and have more control over the environment. To facilitate backup and collaboration, a shared cloud storage (e.g., Google Drive) was used to store model checkpoints, certain evaluation outputs, and large artifacts that would be inconvenient to keep solely under Git.

Version control with Git and a hosted Git repository (e.g., GitHub) was a central resource for coordinating development.All code for training, inference, integration, and evaluation was maintained in this repository, along with configuration files that described model hyperparameters and system settings. This made it possible to recreate specific runs by combining the correct code version with the appropriate checkpoint stored in the shared drive.

The resource plan, overall, was well thought out to meet the needs of GPU-intensive training while taking into account the constraints of academic infrastructure. The main emphasis was placed on software that is open-source and hardware that is readily available to users, rather than on scaling to very large models or datasets. However, it was still ensured that all four components of the multimodal system could be trained, evaluated, and demonstrated.

2.5 Project Schedule

The project schedule for this stage was from late August to the submission date in early December. The work was organized as a sequence of overlapping stages, each one extending and deepening the previous work. Although the schedule was not implemented as a formal Gantt chart in the report, we still followed a very clear timeline which in fact served the same purpose.

The first couple of weeks (late August to early September) were mainly devoted to getting back to the project context, finalizing the scope, and preparing the development environment. During that time, we checked whether the datasets chosen beforehand (ScienceQA, arXiv summarization, and music dataset) were still good for the purpose, made sure that local copies were up to date, and all team members had access to the shared GPU environment and could run basic training and inference scripts. Simultaneously, we went over the requirements again and got a clear understanding of which model variants and features would be handed over at the end of the project.

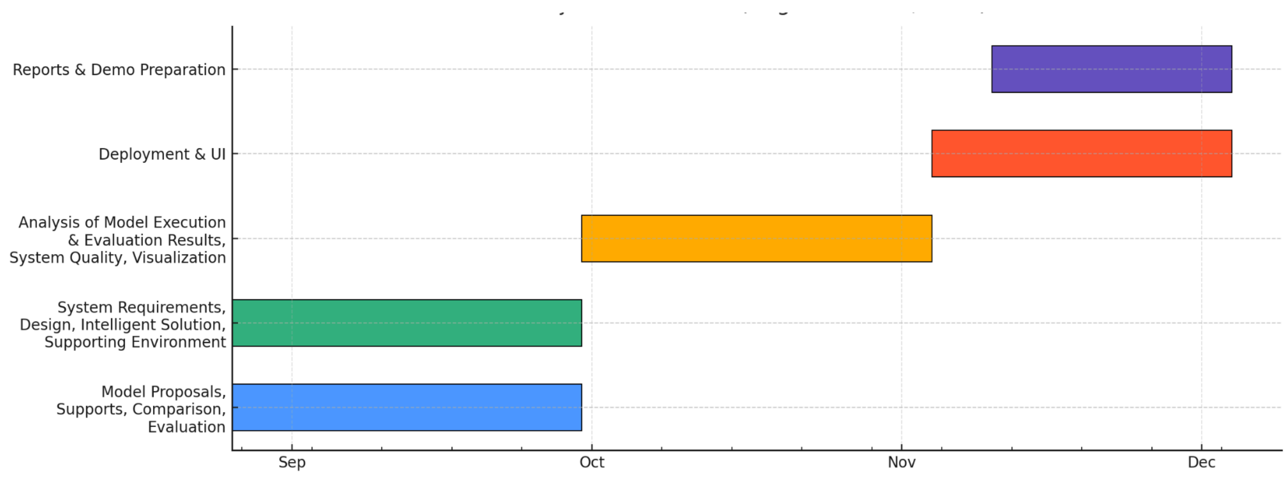
The subsequent phase (around mid-September) was mainly concerned with model pipelines and the first few rounds of fine-tuning. Qwen VLM fine-tuning on ScienceQA and T5-small fine-tuning on the arXiv summarization dataset were prioritized, since these models directly support the educational use cases and involve more complex pipelines (images + text for Qwen, lengthy sequences for T5). Our goal was to have baseline fine-tuned models with stable training curves and preliminary validation metrics by the end of this stage. From late September to early October, we focused on setting up the text-to-music training and evaluation workflow for MusicGen-small.

Once the individual models had worked fine-tuned versions (early to mid-October), the schedule moved into the integration phase. Over several weeks, we implemented and refined backend endpoints for image generation, music generation, multimodal QA, and summarization, and then wired these endpoints into a unified controller. After that, the UI integration stage (late October to early November) followed, where we developed and polished the user interface screens for the social media and education modes, tested end-to-end flows, and got usability feedback and made changes accordingly.

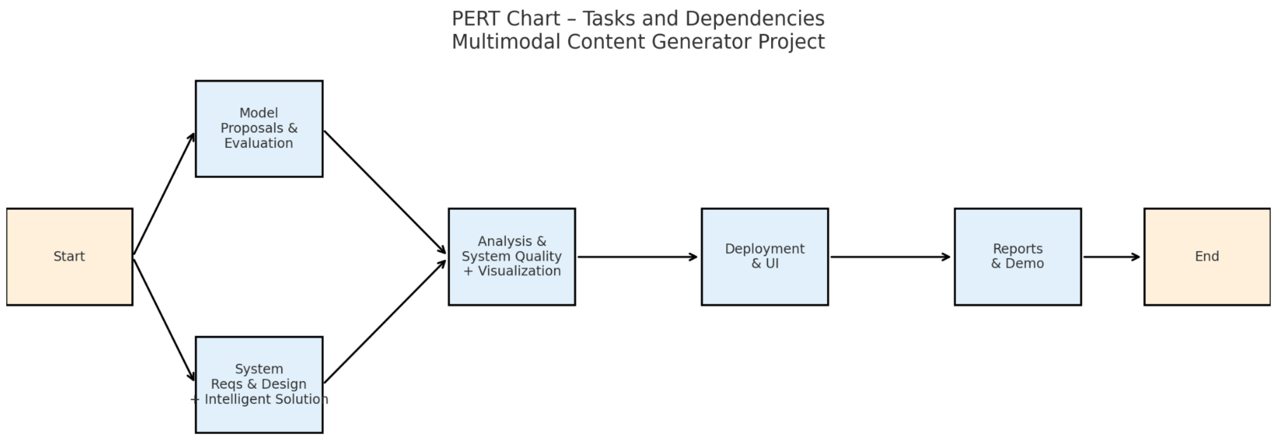
The final stage (mid-November to the submission date in early December) was mainly about evaluation, visualization, and documentation. During the time, we did more systematic evaluation runs for each model, got representative examples for the report and demonstration, and looked into performance issues like latency or occasional poor-quality outputs. Also, we were finishing up the project report, preparing presentation materials, and making sure that all artifacts (code, checkpoints, example outputs) were nicely arranged and backed up.

The staged schedule core model capabilities ensured that they were established early enough for meaningful integration and system-level evaluation, rather than integration being left at the very end. Moreover, it was giving the team checkpoints at the end of each stage to change priorities, deal with unexpected issues, and keep the overall project going.

<u>Gantt Chart:</u>



<u>PERT Chart:</u>



PERT Chart – Tasks and Dependencies
Multimodal Content Generator Project
Start
Model Proposals & Evaluation
System Reqs & Design + Intelligent Solution
Analysis & System Quality + Visualization
Deployment & UI
Reports & Demo
End

3. Data Engineering

3.1 Data Process

The data process for this project was designed to support three core model families—Qwen VLM, T5-small, and musicgen_small—while keeping the flow from raw data to train/validation/test splits as consistent as possible. For each task, we began from publicly available raw datasets and followed a stepped pipeline: (i) acquisition of the original data files, (ii) minimal sanity checks that the raw data matched the expected schema, (iii) preprocessing and cleaning into model-friendly formats, (iv) transformation into standardized dataset objects usable in PyTorch and the Hugging Face ecosystem, and (v) preparation of explicit train, validation, and test splits.

For multimodal QA, the starting point was the ScienceQA dataset, which provides question text, answer options, labels, and optional images. For summarization, we used an arXiv-based scientific text dataset containing long-form input text and shorter target summaries. To generate music, we used a dataset consisting of short audio clips and their corresponding text descriptions or tags that indicated the mood or style. We first downloaded each dataset from its source (e.g., Hugging Face or other open repositories), then we restructured the raw files from each dataset into a uniform directory tree, and finally, we ran the task-specific preprocessing scripts.

After the preprocessing was done, we converted the cleaned data into datasets that were ready for training and also made the splits for training, validation, and test sets. This way, every model could be fine-tuned and tested in line with the project criteria.

3.2 Data Collection

Data collection in this phase relied on curated public datasets accessed via Hugging Face and related tools, rather than ad-hoc scraping. The focus was on assembling and organizing benchmark datasets suitable for multimodal science question answering, scientific text summarization, and music generation, and on making their splits and basic statistics explicit.

ScienceQA (for Qwen VLM)

For multimodal science question answering, we used the ScienceQA dataset as mirrored on Hugging Face under the path hf://datasets/derek-thomas/ScienceQA/, as shown in ScienceQA_Data_Preprocessing_EDA.ipynb. Three parquet files corresponding to the official splits are loaded:

- data/train-00000-of-00001-1028f23e353fbe3e.parquet
- data/validation-00000-of-00001-6c7328ff6c84284c.parquet
- data/test-00000-of-00001-f0e719df791966ff.parquet

These are concatenated into a single DataFrame df_raw with:

- Total questions: 21,208

A summary statistics table computed in the notebook reports the following counts and properties:

- Questions with text context (hint): 10,220
- Questions with image context: 10,332
- Questions with both text and image context: 6,532
- Questions without any additional context: 7,188
- Questions with a lecture field: 17,798

- Questions with an explanation field: 19,202
- Distinct question texts: 9,122
- Distinct lecture texts: 262
- Number of topic classes: 26
- Number of category classes: 127
- Number of skill classes: 379

Average text lengths (in tokens/words as computed in the notebook) are:

- Average question length: 12.11
- Average choice length: 4.78
- Average lecture length: 104.95
- Average explanation length: 43.15

Each record contains: the question text, a list of answer choices, the correct label, optional lecture and explanation text, and an optional image (stored via a path into the ScienceQA image store). The notebook also prints a small sample (first three rows) showing the raw structure of the dataset.

arXiv Summarization Corpus (for T5-small)

For scientific text summarization, we used an arXiv-based summarization corpus organized as a Hugging Face DatasetDict stored locally under arxiv_data/hf_splits, as shown in Arxix_Data_Preprocessing_EDA.ipynb. The dataset is loaded with:

```
from datasets import load_from_disk
```

```
dataset = load_from_disk("arxiv_data/hf_splits")

train_ds = dataset["train"]

val_ds   = dataset["validation"]

test_ds  = dataset["test"]
```

The notebook prints the following raw split sizes:

- Train (raw): Shape (203,037, 2) → 203,037 examples, 2 columns (article, abstract)
- Validation (raw): Shape (6,436, 2)
- Test (raw): Shape (6,440, 2)

After computing character-length features for both article and summary and applying any cleaning logic encoded in the notebook, the cleaned training split has:

- Train (clean): Shape (202,596, 4) → columns article, abstract, article_char_len, summary_char_len

The validation and test splits are used in their original two-column form (article, abstract) in the notebook outputs, with missing-value checks confirming that there are no NaN entries in these fields. Each record therefore consists of a long-form article text (one or more paragraphs) and a shorter human-written abstract, which serves as the target summary for training and evaluating T5-small.

In addition, Arxiv_Data_Collection.ipynb demonstrates a smaller custom arXiv sampling pipeline using the arxiv Python API. In that notebook:

- 100 records are fetched from arXiv via API search.
- The resulting DataFrame is then split with approximate proportions of 94% / 3% / 3%, yielding:
    - Train: 94 examples
    - Validation: 3 examples
    - Test: 3 examples

These 100 examples are stored in different splits under arxiv_data/splits and have been converted into a small Hugging Face dataset under arxiv_data/hf_splits. This reduced subset is primarily utilized for testing the completion and splitting pipeline, while the principal summarization experiments are funded by the substantially larger 203k-example dataset mentioned above.

Music Dataset (for musicgen_small)

For music generation, Data_Cleaning_and_analysis.ipynb loads the GTZAN genre classification dataset via Hugging Face (sanchit-gandhi/gtzan). The notebook maps over all examples, extracts basic audio features (such as duration and loudness), and constructs a cleaned DataFrame with one row per audio clip. The following counts are printed:

- Original samples: 999
- Clean samples: 999

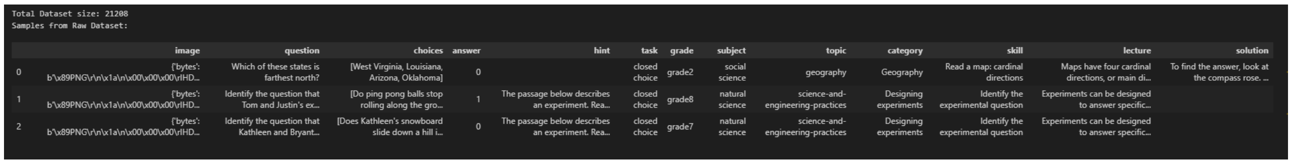
This confirms that all 999 clips from the dataset are successfully processed and retained. Each record includes:

- The file path to the audio clip
- A genre label (integer-coded)

- Derived features such as duration_sec, rms, pitch_mean, and mel_energy

These genre labels and derived features are used for exploratory analysis (e.g., genre distribution, duration distribution, and example mel-spectrograms), and they provide structured style descriptors that can be aligned with text prompts when fine-tuning musicgen_small for social media music generation.

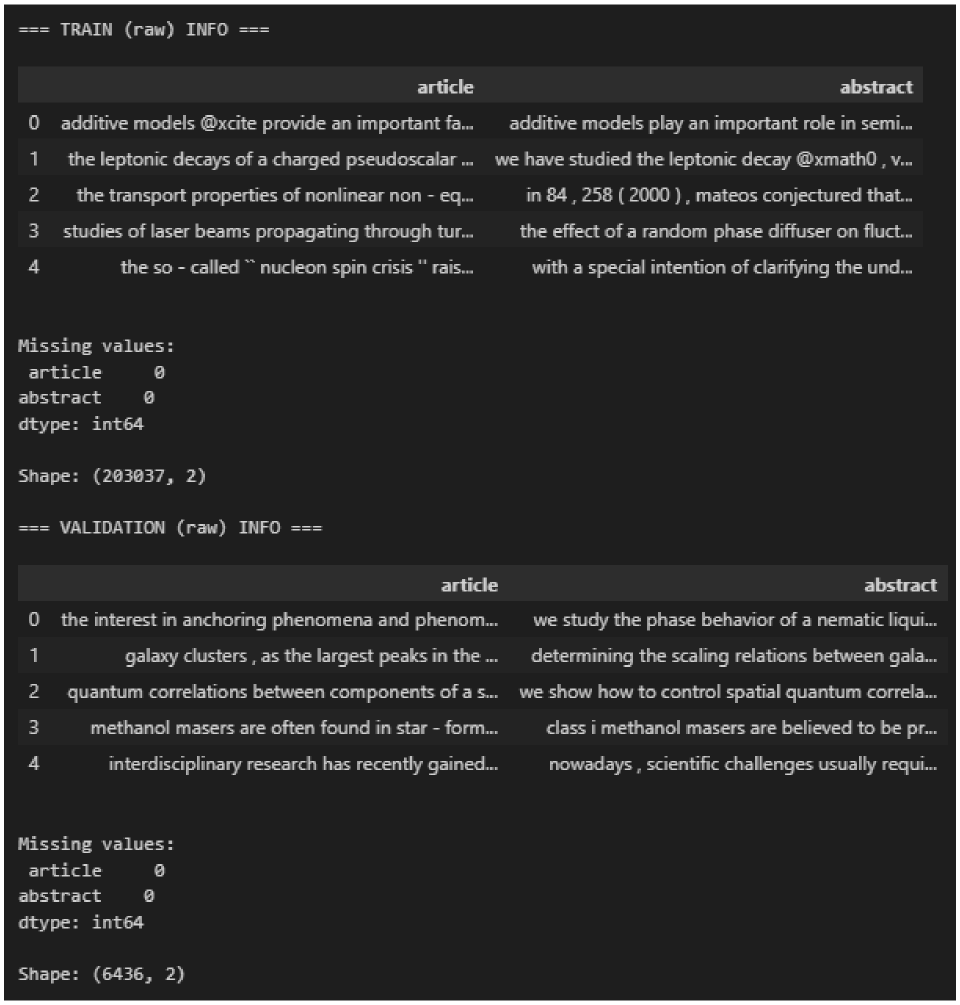
ScienceQA:



```
Total Dataset size: 21208
Samples from Raw Dataset:
```

	image	question	choices	answer	hint	task	grade	subject	topic	category	skill	lecture	solution
0	{'bytes': b'\x89PNG\r\n\x1a\n\x00\x00\x00\rIHD...	Which of these states is farthest north?	[West Virginia, Louisiana, Arizona, Oklahoma]	0		closed choice	grade2	social science	geography	Geography	Read a map: cardinal directions	Maps have four cardinal directions, or main di...	To find the answer, look at the compass rose. ...
1	{'bytes': b'\x89PNG\r\n\x1a\n\x00\x00\x00\rIHD...	Identify the question that Tom and Justin's ex...	[Do ping pong balls stop rolling along the gro...	1	The passage below describes an experiment. Rea...	closed choice	grade8	natural science	science-and-engineering-practices	Designing experiments	Identify the experimental question	Experiments can be designed to answer specific...	
2	{'bytes': b'\x89PNG\r\n\x1a\n\x00\x00\x00\rIHD...	Identify the question that Kathleen and Bryant...	[Does Kathleen's snowboard slide down a hill i...	0	The passage below describes an experiment. Rea...	closed choice	grade7	natural science	science-and-engineering-practices	Designing experiments	Identify the experimental question	Experiments can be designed to answer specific...	

arXiv:



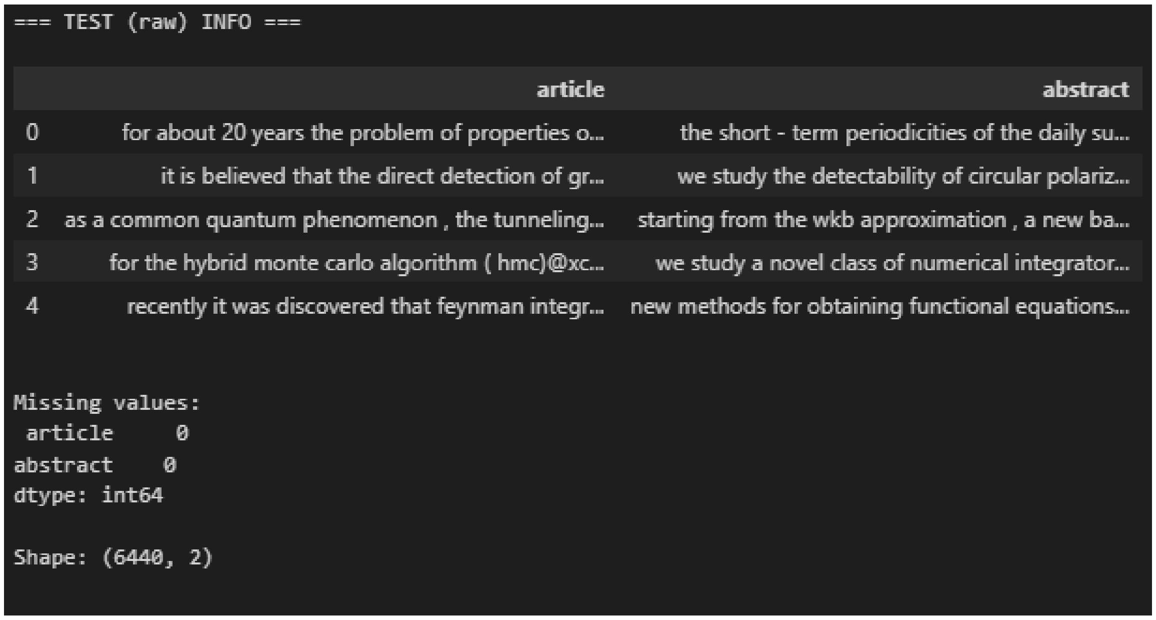
```
=== TRAIN (raw) INFO ===
```

	article	abstract
0	additive models @xcite provide an important fa...	additive models play an important role in semi...
1	the leptonic decays of a charged pseudoscalar ...	we have studied the leptonic decay @xmath0 , v...
2	the transport properties of nonlinear non - eq...	in 84 , 258 (2000) , mateos conjectured that...
3	studies of laser beams propagating through tur...	the effect of a random phase diffuser on fluct...
4	the so - called `` nucleon spin crisis '' rais...	with a special intention of clarifying the und...

```
Missing values:
 article     0
abstract    0
dtype: int64

Shape: (203037, 2)

=== VALIDATION (raw) INFO ===
```

	article	abstract
0	the interest in anchoring phenomena and phenom...	we study the phase behavior of a nematic liqui...
1	galaxy clusters , as the largest peaks in the ...	determining the scaling relations between gala...
2	quantum correlations between components of a s...	we show how to control spatial quantum correla...
3	methanol masers are often found in star - form...	class i methanol masers are believed to be pr...
4	interdisciplinary research has recently gained...	nowadays , scientific challenges usually requi...

```
Missing values:
 article     0
abstract    0
dtype: int64

Shape: (6436, 2)
```



```
=== TEST (raw) INFO ===
```

	article	abstract
0	for about 20 years the problem of properties o...	the short - term periodicities of the daily su...
1	it is believed that the direct detection of gr...	we study the detectability of circular polariz...
2	as a common quantum phenomenon , the tunneling...	starting from the wkb approximation , a new ba...
3	for the hybrid monte carlo algorithm (hmc)@xc...	we study a novel class of numerical integrator...
4	recently it was discovered that feynman integr...	new methods for obtaining functional equations...

```
Missing values:
 article     0
abstract    0
dtype: int64

Shape: (6440, 2)
```

3.3 Data Pre-processing

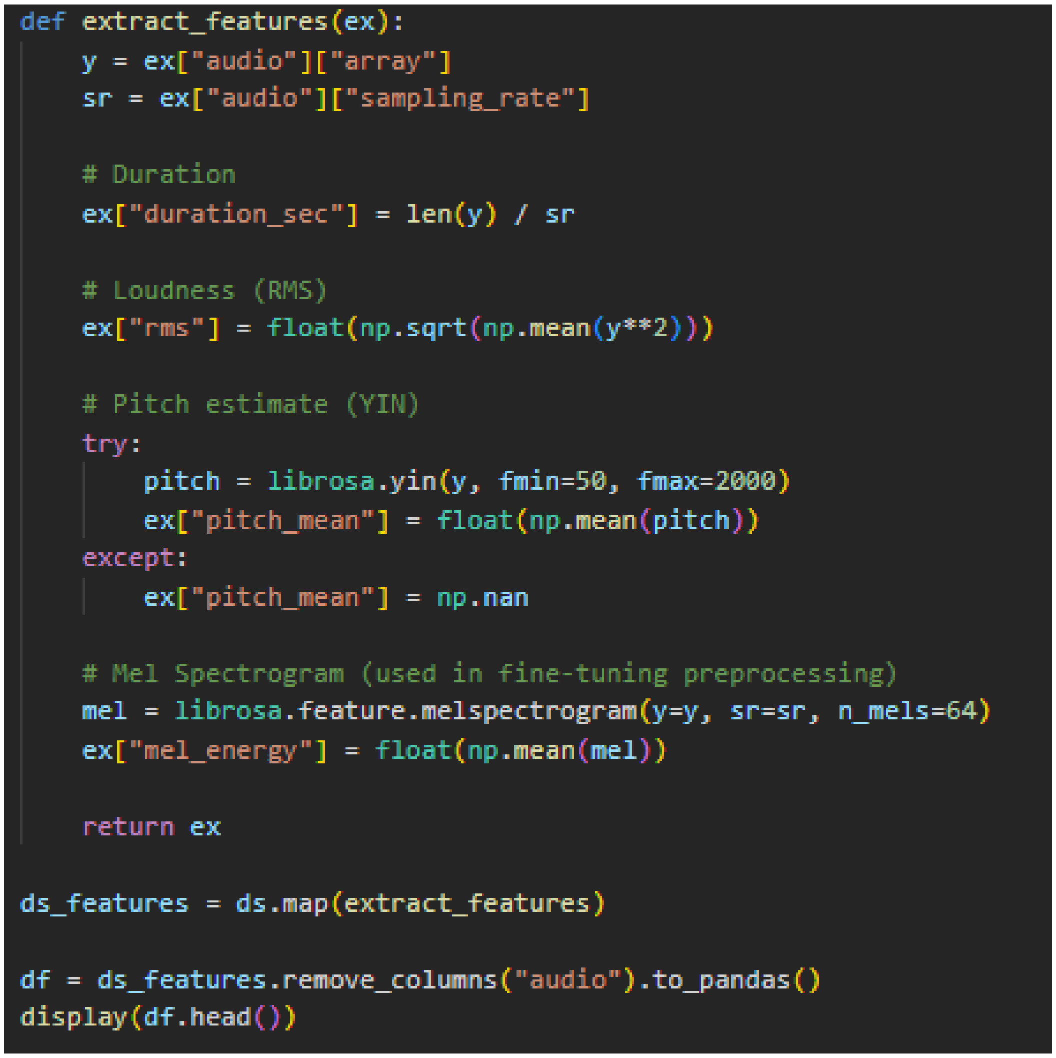
After collection, each dataset underwent task-specific preprocessing to ensure that the data was clean, consistent, and directly consumable by the models.

For ScienceQA, preprocessing began by parsing the original data structure into the fields required by Qwen VLM: the image path or image object (when available), the question text, the set of answer options, and the correct label. Image files were loaded and converted to a standard RGB format, and any missing or corrupted images were filtered out or handled carefully. We then applied the official Qwen processor to resize and normalize images, and to tokenize the concatenated question and options into input IDs and attention masks. The result was a set of multimodal examples that Qwen VLM could directly consume during fine-tuning.

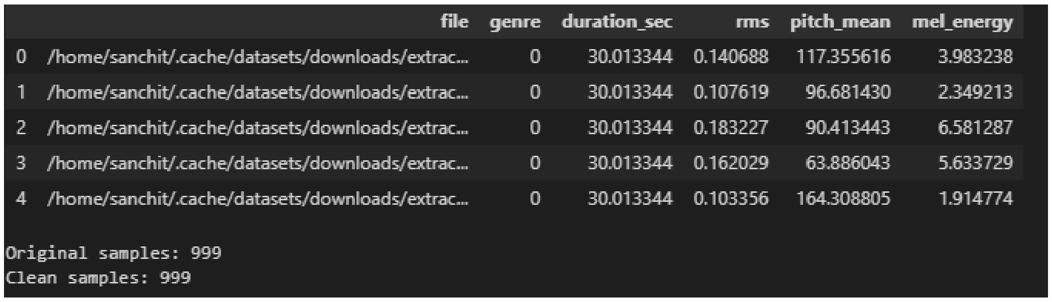
For the arXiv summarization dataset, preprocessing focused on cleaning and normalizing the text. We extracted the input text and corresponding summaries, removed obvious artifacts such as stray LaTeX markup (where possible), and normalized whitespace and encoding. Using the T5 tokenizer, we tokenized both the source text and the target summary while enforcing maximum sequence lengths on inputs and outputs to keep training efficient and prevent memory issues. Entries with extremely short or clearly malformed text were either cleaned or, if necessary, filtered out.

For the music dataset, preprocessing steps operated at the audio level. Audio clips were resampled (if needed) to a consistent sample rate and converted to a standard format. Clips were trimmed or padded to a target duration suitable for social media usage, ensuring that the model saw relatively uniform-length examples. The accompanying text prompts or tags were cleaned (e.g., lowercasing, basic normalization) and stored alongside the audio tensors.

GTZAN feature extraction snippet for audio clips:



```python
def extract_features(ex):
    y = ex["audio"]["array"]
    sr = ex["audio"]["sampling_rate"]

    # Duration
    ex["duration_sec"] = len(y) / sr

    # Loudness (RMS)
    ex["rms"] = float(np.sqrt(np.mean(y**2)))

    # Pitch estimate (YIN)
    try:
        pitch = librosa.yin(y, fmin=50, fmax=2000)
        ex["pitch_mean"] = float(np.mean(pitch))
    except:
        ex["pitch_mean"] = np.nan

    # Mel Spectrogram (used in fine-tuning preprocessing)
    mel = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=64)
    ex["mel_energy"] = float(np.mean(mel))

    return ex

ds_features = ds.map(extract_features)

df = ds_features.remove_columns("audio").to_pandas()
display(df.head())
```



	file	genre	duration_sec	rms	pitch_mean	mel_energy
0	/home/sanchit/.cache/datasets/downloads/extrac...	0	30.013344	0.140688	117.355616	3.983238
1	/home/sanchit/.cache/datasets/downloads/extrac...	0	30.013344	0.107619	96.681430	2.349213
2	/home/sanchit/.cache/datasets/downloads/extrac...	0	30.013344	0.183227	90.413443	6.581287
3	/home/sanchit/.cache/datasets/downloads/extrac...	0	30.013344	0.162029	63.886043	5.633729
4	/home/sanchit/.cache/datasets/downloads/extrac...	0	30.013344	0.103356	164.308805	1.914774

```
Original samples: 999
Clean samples: 999
```

3.4 Data Transformation

Once preprocessing was complete, we transformed the cleaned data into formats that integrate smoothly with our training and inference pipelines. For all three tasks, this meant wrapping the data into either Hugging Face Datasets objects or custom PyTorch Dataset classes.

For ScienceQA, each example was represented as a multimodal record containing processed pixel values for the image, tokenized text (input IDs and attention masks) for the question and options, and a label representing the correct answer. These records were bundled into a dataset object with an associated collate function that uses the Qwen processor to pad sequences and stack image tensors into uniform batches. This ensured that batches passed to Qwen VLM were correctly aligned and ready for training or evaluation.

For the arXiv summarization dataset, we created dataset objects where each entry consists of tokenized input text (input IDs and attention masks) and tokenized target summaries (labels). These were also paired with a collate function that leverages the T5 tokenizer's padding utilities to build uniformly shaped batches. This transformation step allowed us to plug the dataset directly into the T5-small training loop without additional ad hoc preprocessing.

For the music data, entries were structured as a pair of an audio tensor (or encoded representation) and its associated text prompt. These were wrapped into a dataset class that feeds directly into the MusicGen fine-tuning code. Batching handled consistent audio shapes (based on the pre-defined clip length) and tokenized text prompts.

3.5 Data Preparation

Data preparation focused on constructing clear and reproducible train, validation, and test splits from the transformed datasets so that model performance could be assessed fairly and consistently across experiments. For each task, we followed the general pattern of using the official training and validation splits when they were provided, and otherwise partitioning the data into subsets that respected task constraints.

For ScienceQA, the intent was to keep evaluation strictly disjoint from training by separating questions into training, validation, and (if used) test sets based on the dataset's existing split definitions. The training split was used for fine-tuning Qwen VLM, while the validation split was used to monitor accuracy or other metrics and to guide decisions such as when to stop training or adjust hyperparameters.

For the arXiv summarization dataset, the prepared training set contained the majority of the examples, while a smaller validation set was used to track validation loss and summarization quality. If a dedicated test split was provided or created, it was reserved for final evaluation runs to avoid overfitting to the validation data. Each split was stored as a separate dataset object (or a tagged subset) so that training scripts could load them explicitly and reproducibly.

For the music dataset, the training split contained most of the audio–text pairs, and a smaller validation split was used to check for overfitting and to subjectively evaluate generation quality from held-out prompts.

3.6 Data Statistics

To better understand the datasets and support modeling decisions, we computed descriptive statistics at different stages of the data pipeline. For ScienceQA, this included statistics such as the number of questions after preprocessing, the proportion of questions with images versus text-only questions, and the distribution of question categories when such labels were available.

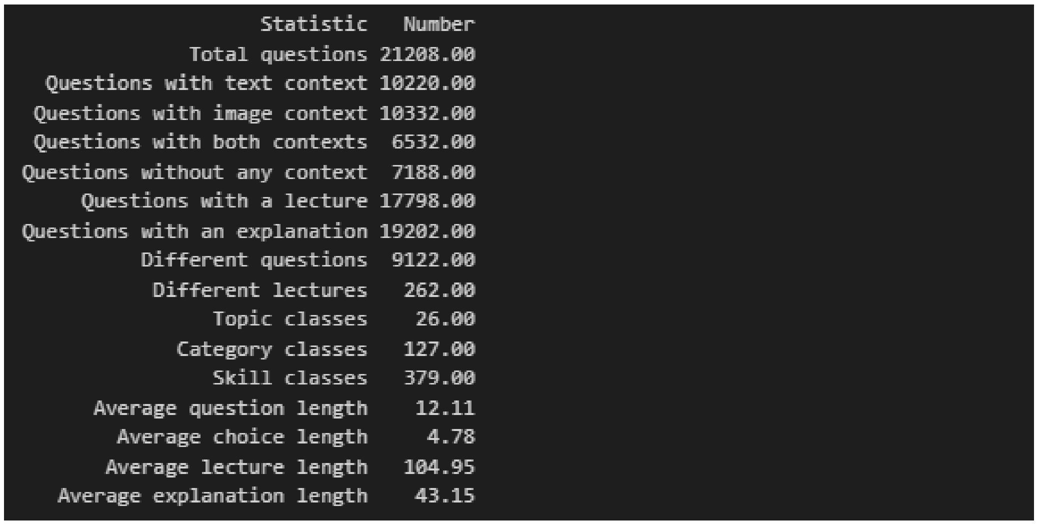
These statistics helped confirm that the dataset remained reasonably balanced across the types of science questions we expected Qwen VLM to handle.

For the arXiv summarization dataset, we examined the distribution of input text lengths (in tokens or characters) and summary lengths, as well as basic properties such as the frequency of different scientific domains (when identifiable). Length statistics were particularly important for setting maximum sequence lengths for T5-small and for understanding how aggressively long inputs needed to be truncated. We also checked for extreme outliers (very short or extremely long texts) to avoid destabilizing training.

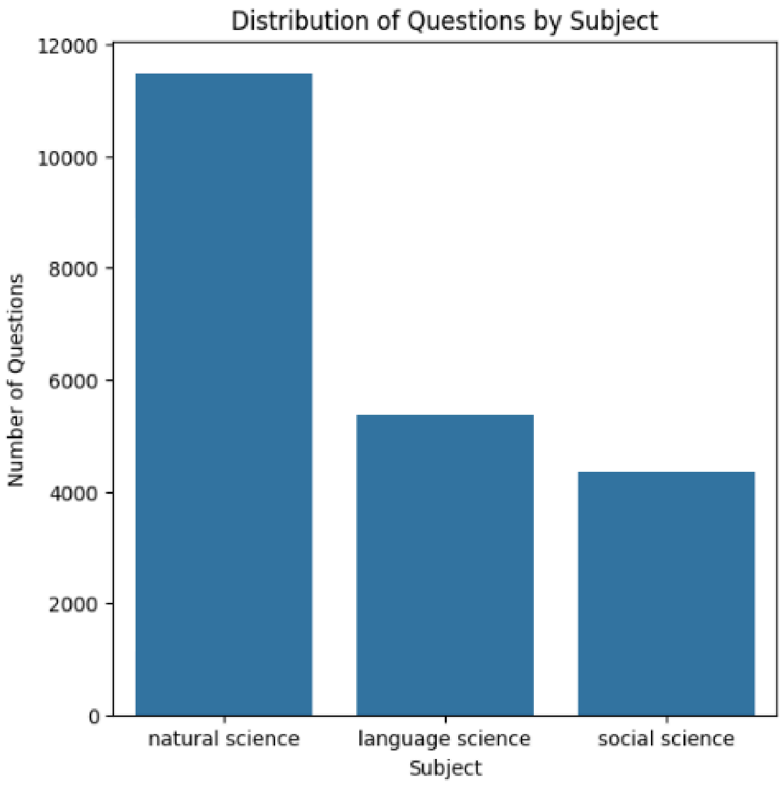
For the music dataset, we considered properties such as the distribution of audio clip durations (after trimming/padding) and the distribution of tag types (e.g., how many clips were labeled as "lofi", "cinematic", etc., if available). This provided a sense of how varied the style/mood space was and whether the data coverage matched the intended social media use cases.

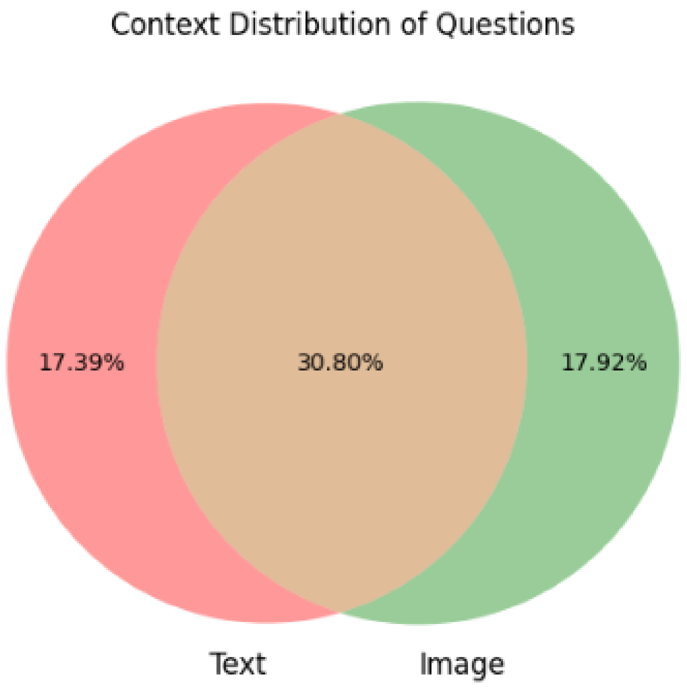
In the analysis notebooks, these statistics were typically summarized using simple tables and visualizations such as histograms or bar charts.

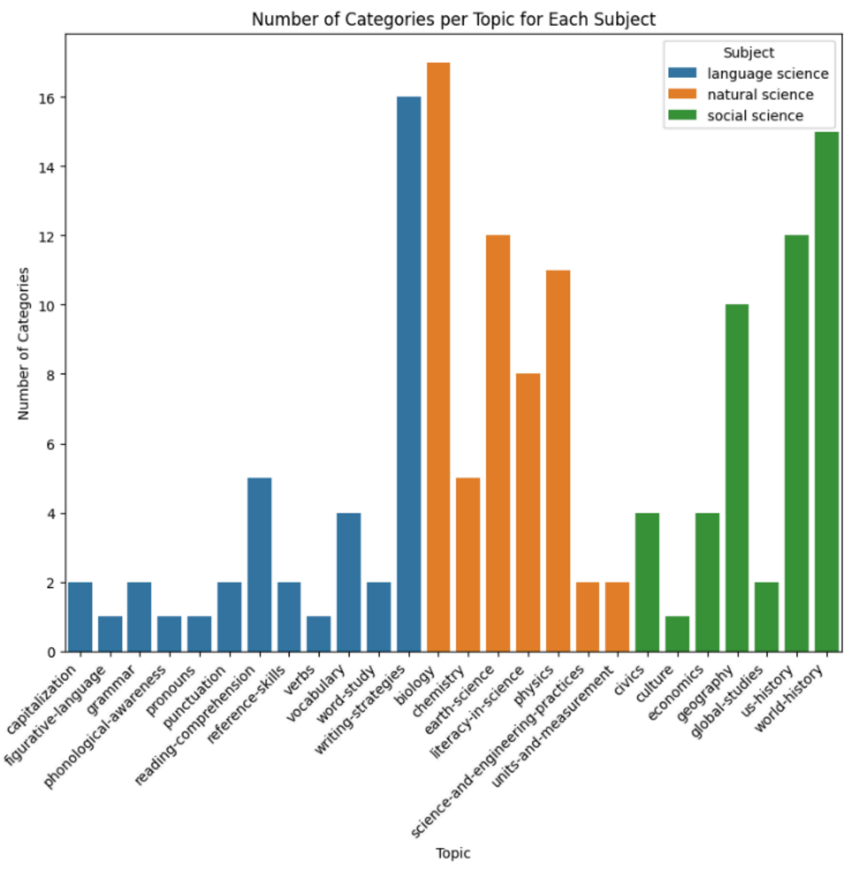
ScienceQA Statistics:

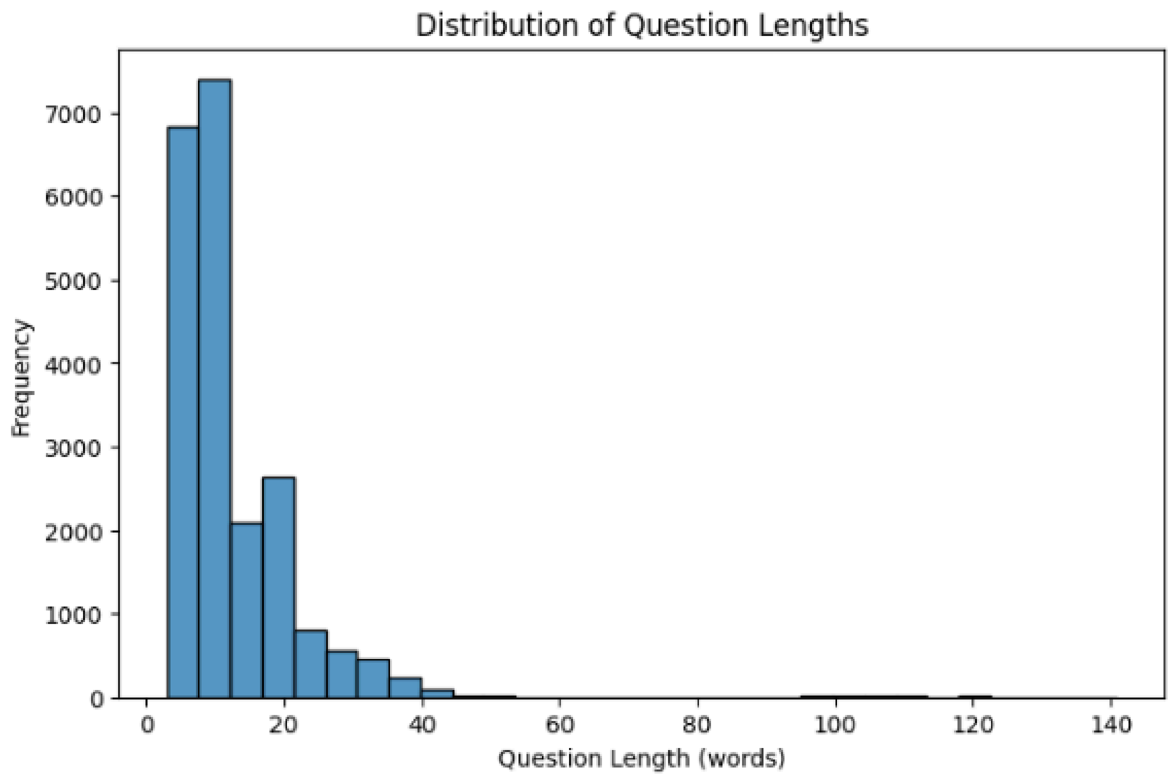


```
                  Statistic   Number
            Total questions 21208.00
 Questions with text context 10220.00
Questions with image context 10332.00
Questions with both contexts  6532.00
Questions without any context  7188.00
     Questions with a lecture 17798.00
Questions with an explanation 19202.00
          Different questions  9122.00
           Different lectures   262.00
               Topic classes    26.00
            Category classes   127.00
               Skill classes   379.00
     Average question length    12.11
       Average choice length     4.78
      Average lecture length   104.95
  Average explanation length    43.15
```

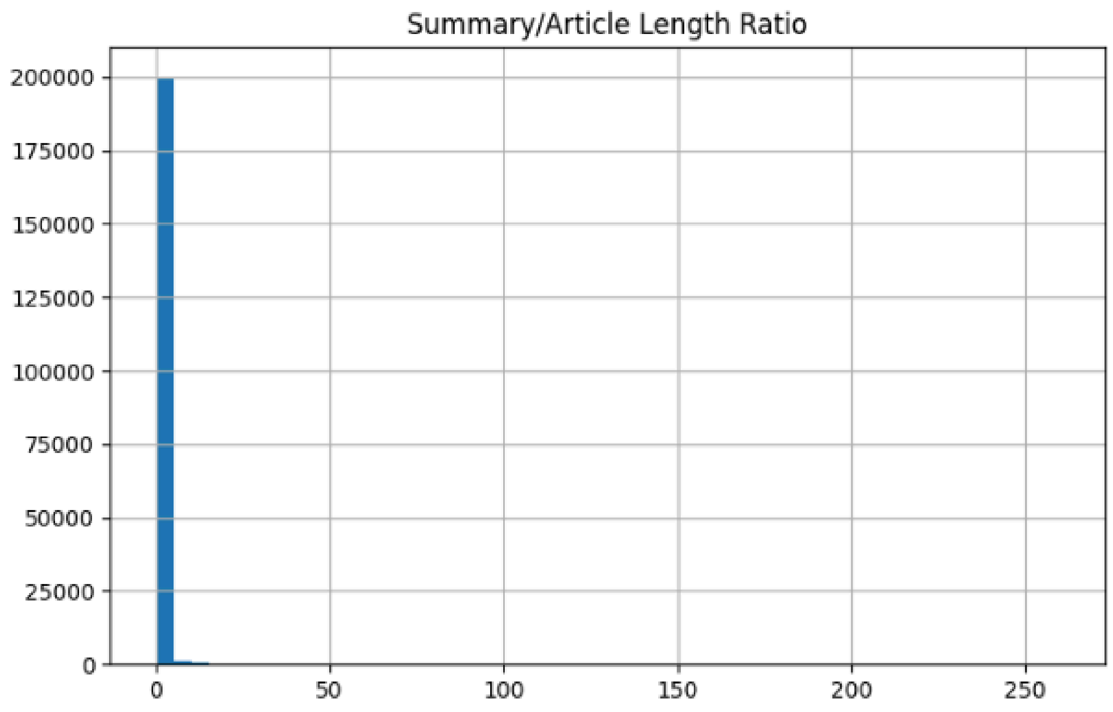


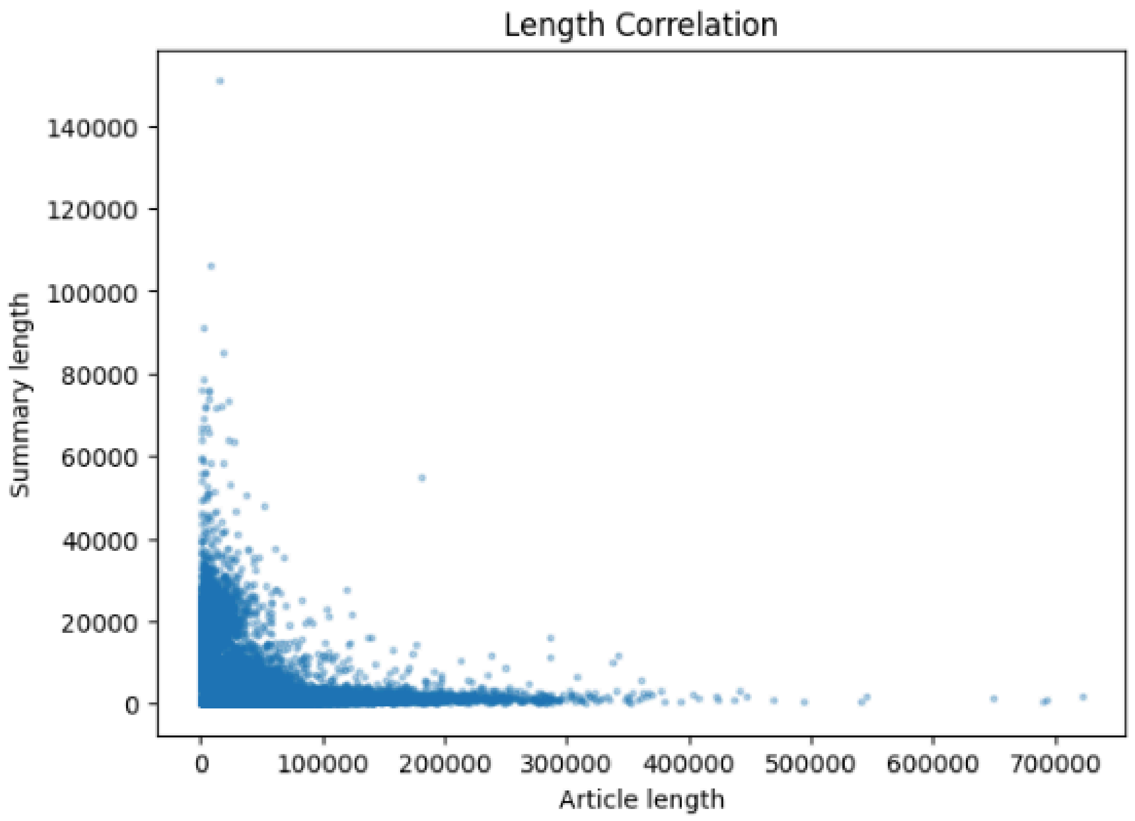

Context Distribution of Questions
17.39%
30.80%
17.92%
Text
Image

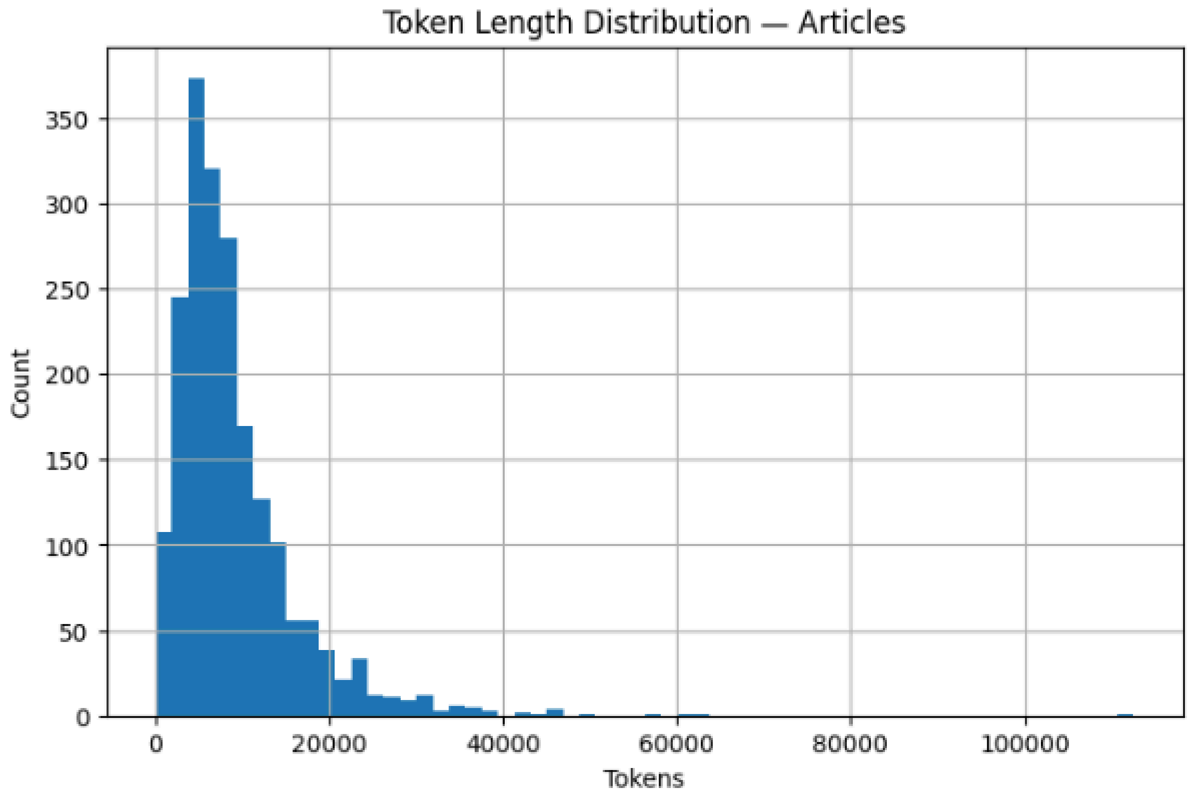

Number of Categories per Topic for Each Subject
Subject
language science
natural science
social science
Number of Categories
Topic
capitalization
figurative-language
grammar
phonological-awareness
pronouns
punctuation
reading-comprehension
reference-skills
verbs
vocabulary
word-study
writing-strategies
biology
chemistry
earth-science
literacy-in-science
physics
science-and-engineering-practices
units-and-measurement
civics
culture
economics
geography
global-studies
us-history
world-history

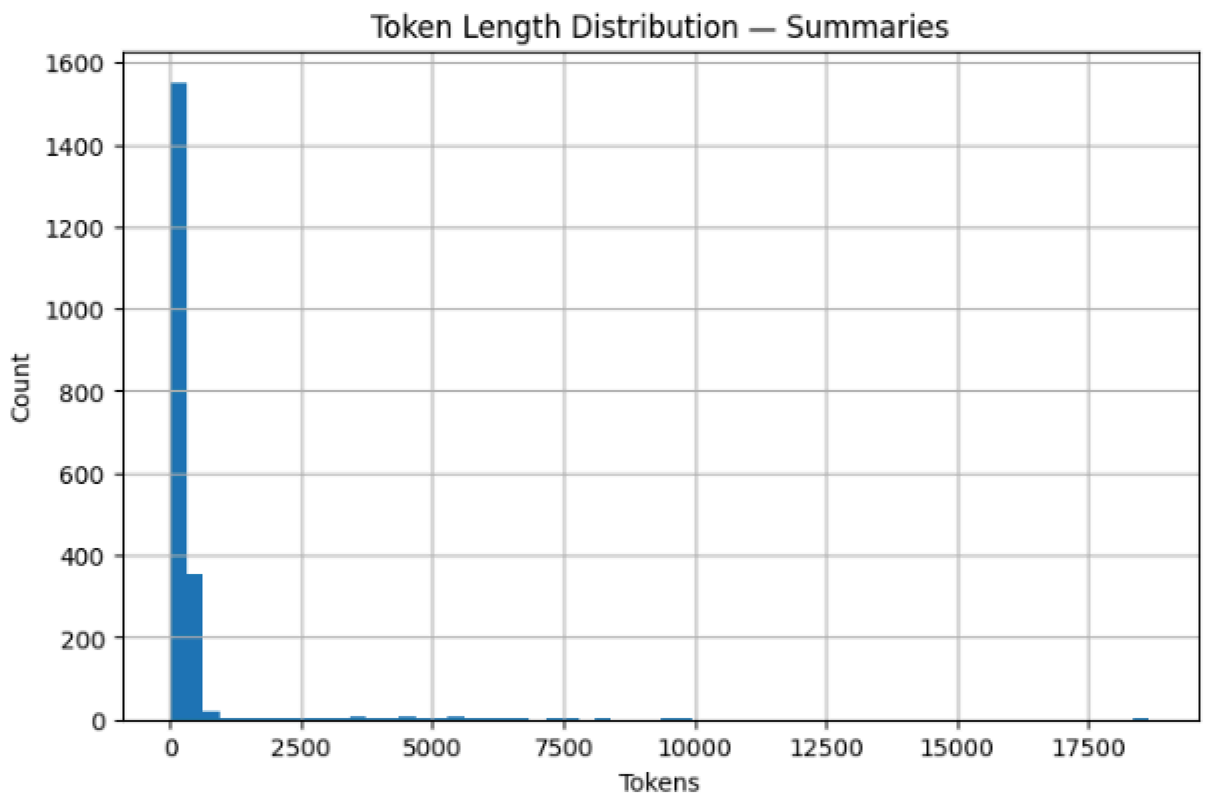

Distribution of Question Lengths
Frequency
Question Length (words)
0
1000
2000
3000
4000
5000
6000
7000
0
20
40
60
80
100
120
140

arXiv:

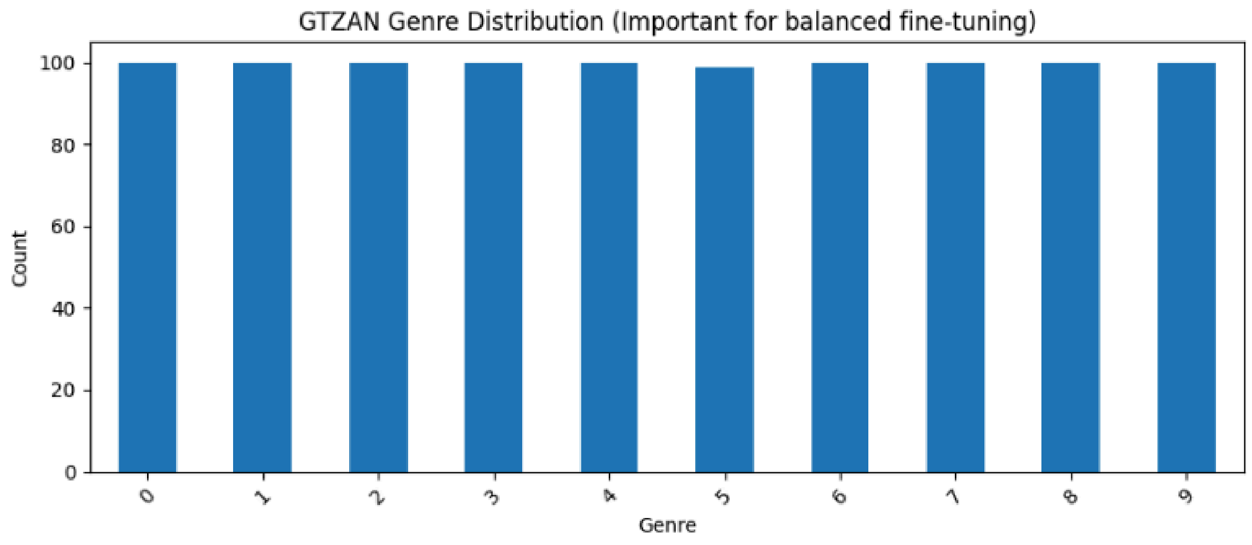

Summary/Article Length Ratio
0
25000
50000
75000
100000
125000
150000
175000
200000
0
50
100
150
200
250


Length Correlation
Summary length
Article length
0
20000
40000
60000
80000
100000
120000
140000
0
100000
200000
300000
400000
500000
600000
700000


Token Length Distribution — Articles
Count
Tokens
0
50
100
150
200
250
300
350
0
20000
40000
60000
80000
100000


Token Length Distribution — Summaries
Count
Tokens
1600
1400
1200
1000
800
600
400
200
0
0
2500
5000
7500
10000
12500
15000
17500

GTZAN:


GTZAN Genre Distribution (Important for balanced fine-tuning)
Count
Genre
100
80
60
40
20
0
0
1
2
3
4
5
6
7
8
9

3.7 Data Analytics Results

Beyond basic descriptive statistics, we also used data analytics to gain higher-level insights that influenced modeling and system design choices. For ScienceQA, analyzing the distribution of question types and the proportion of questions with images versus text-only helped confirm that fine-tuning Qwen VLM on this dataset would meaningfully exercise both its visual and textual capabilities. It also highlighted any categories that might be particularly challenging or underrepresented, which can be considered when interpreting evaluation results.

For the arXiv summarization dataset, analytics focused on understanding how the dataset aligns with the intended use case of summarizing research-like content for students and researchers. By examining the range of subject areas, the typical structure of input text (e.g., abstracts versus longer sections), and the relationship between input and summary lengths, we confirmed that the data is suitable for training T5-small to produce concise, informative summaries. Visualizations such as scatter plots or length comparison charts (input vs. summary) help illustrate how aggressively the model needs to compress long inputs.

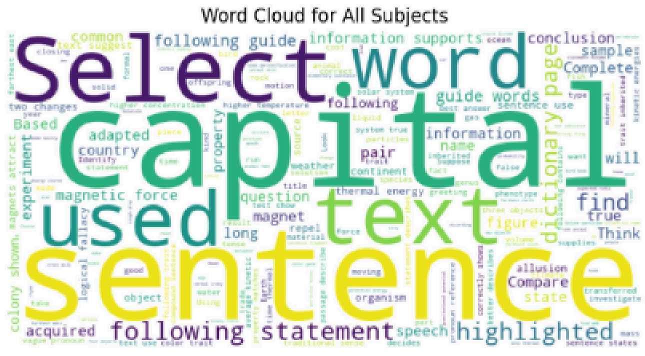
In the music dataset, we used analytics to explore the relationship between textual descriptors (mood/style tags) and the distribution of clips. As an instance, we can foresee the refined musicgen_small model to work excellently for those prompts that use the mentioned styles based on the number of their occurrences in the dataset.

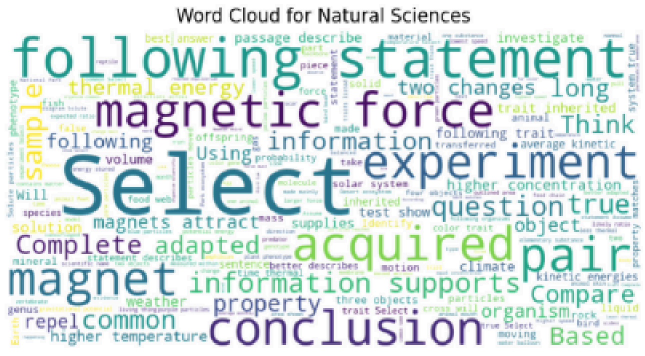
On the other hand, discovering underrepresented tags may be useful for determining the performance of the model and deciding support for data augmentation or collection in the case of a continuation of the project.

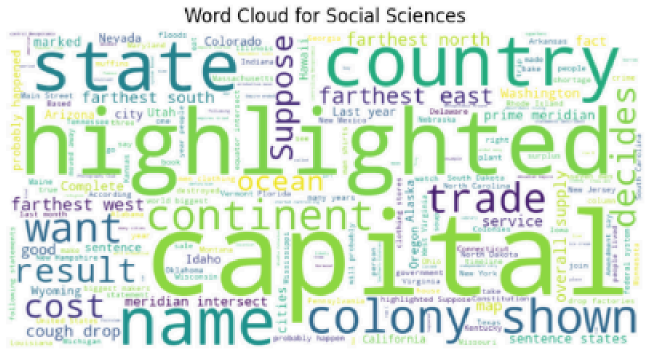
We considered these analytics outcomes as a roadmap to our understanding of the model output and the presence of the system's strengths and weaknesses.

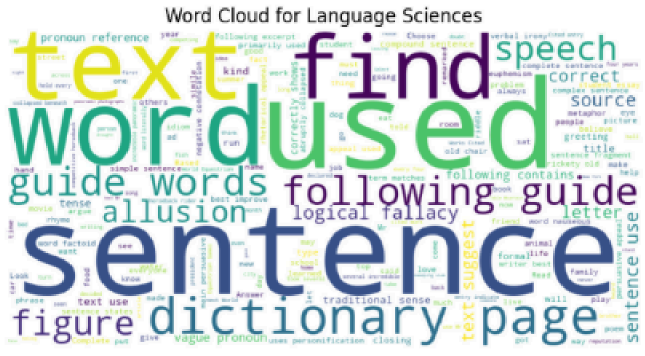
Firstly, they explain the context for any biases or failure modes that have been noticed, thus, linking them back to the data instead of treating them as issues solely at the model level.

<u>ScienceQA:</u>

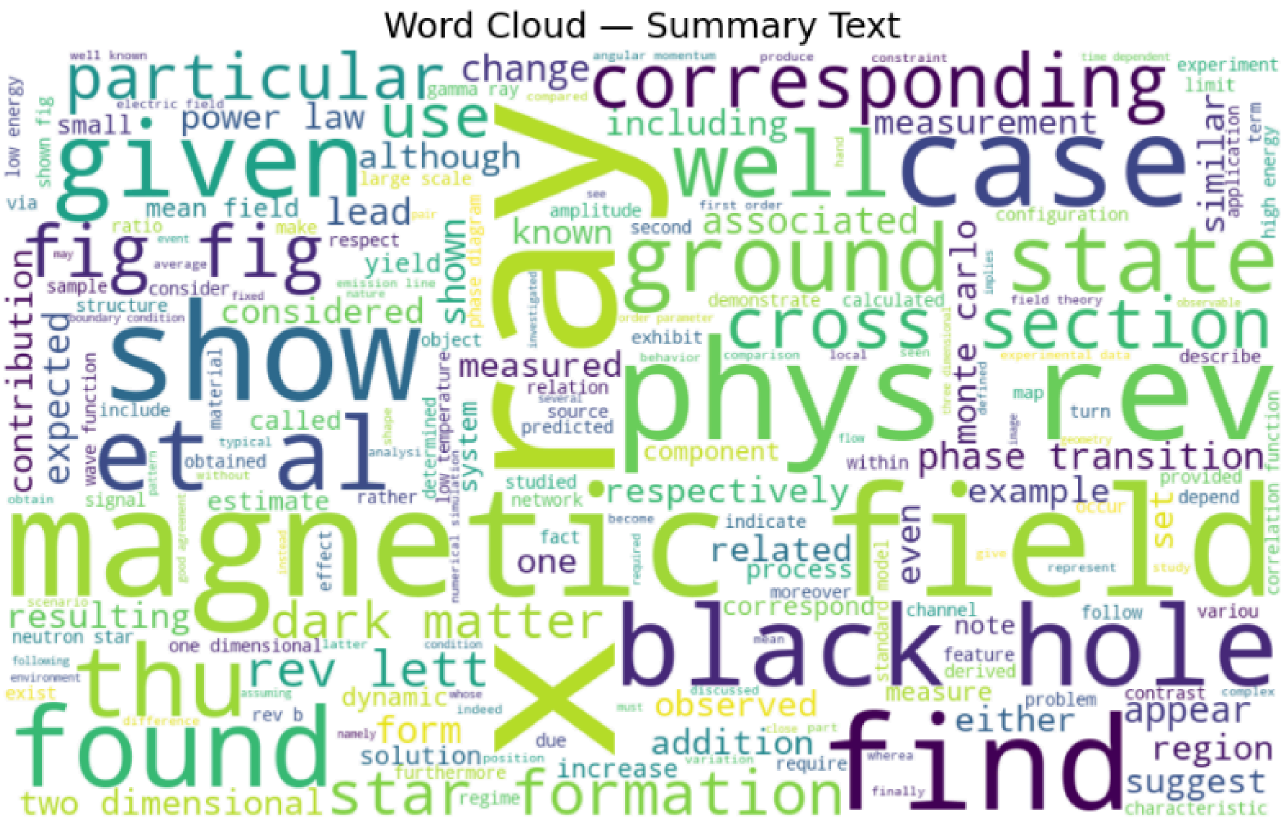




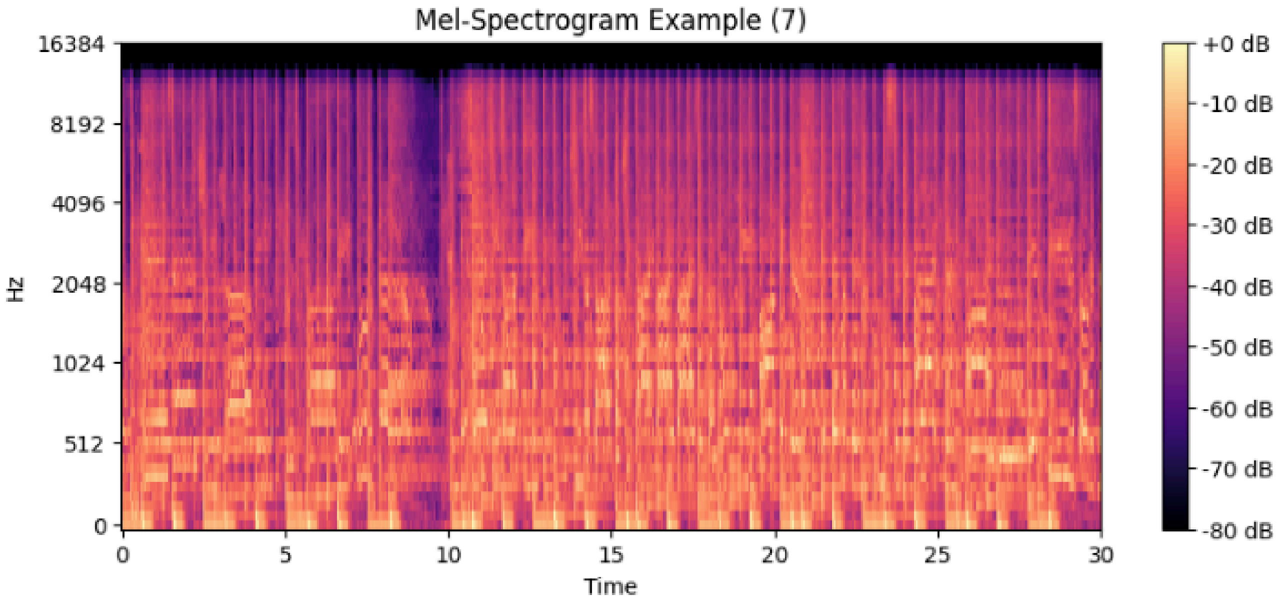




arXiv:


Word Cloud — Summary Text
x ray
magnetic field
black hole
phys rev
ground state
show
given
case
corresponding
find
found
et al
cross section
well
particular
fig
thu
star formation
dark matter
rev lett
phase transition
monte carlo
respectively
example
associated
use
change
power law
although
including
measurement
similar
considered
measured
contribution
expected
two dimensional
observed
addition
increase
suggest
region
appear
either
related
process
one
known
lead
mean field
yield
resulting
neutron star
one dimensional
form
solution
estimate
called
wave function
system
low temperature
component
amplitude
configuration
demonstrate
calculated
exhibit
describe
map
turn
depend
correspond
note
feature
derived
channel
follow
contrast
problem
characteristic
high energy
correlation function
experiment
limit
gamma ray
large scale
small
electric field
well known
angular momentum
produce
constraint
time dependent
first order
second
application
term
low energy
shown fig
via
ratio
make
respect
sample
consider
structure
boundary condition
emission line
phase diagram
object
include
material
typical
obtained
signal
rather
studied
network
fact
indicate
moreover
even
set
model
standard
measure
furthermore
regime
due
rev b
dynamic
effect
determined
source
predicted
relation
within
provided
variou
occur
represent
study
complex
exist
scenario
following
environment
difference
namely
indeed
finally
whereas
require
variation
position
discussed
close
become
see
several
order parameter
field theory
observable
experimental data
comparison
behavior
local
seen
three dimensional
defined
image
flow
geometry
implies
investigated
numerical simulation
good agreement
instead
latter
condition
whose
must
mean
assuming
average
may
fixed
nature
pair
compared
hand
event
obtain
without
pattern
analysis
shape
give
required

GTZAN:


Mel-Spectrogram Example (7)
Hz
16384
8192
4096
2048
1024
512
0
Time
0
5
10
15
20
25
30
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB

4. Model Development

4.1 Model Proposals (Problem Decomposition → Model Map)

User Task	Problem Type	Model Used	Input	Output
Social media image generation	Text → Image	Stable Diffusion	Text prompt	Generated image(s)
Social media music generation	Text → Audio (music)	musicgen_small (FT)	Style/mood + optional params	Audio clip
Science multimodal QA	Image + Text → Answer	Qwen VLM (FT on ScienceQA)	Question text + image	Answer (option / explanation)

Text summarization (papers, etc.)	Long Text → Short Summary	T5-small (FT on arXiv)	Long text	Summarized text

High-level mapping from user tasks to models:For the visual question answering task, we propose a fine-tuned Qwen2-VL model that treats ScienceQA problems as multimodal multiple-choice questions. Each example consists of an image (diagram, chart or picture), a natural language question, a set of answer options and optionally a hint or lecture text. In Qwen_finetuned.ipynb, these elements are mapped into Qwen's chat-style multimodal format. The system message fixes the role of the model as a helpful science tutor. The user message includes the image and a formatted block that contains the question, the labelled answer choices and any available hint or lecture. The assistant message is used as the training target and is standardized to begin with the correct option letter followed by a short explanation, for example "C. The object is accelerating because…". Conceptually, the model takes combined image and text as input and produces a free-form explanation that encodes both the chosen option and a justification. The underlying architecture is Qwen/Qwen2-VL-2B-Instruct, a vision–language transformer that internally encodes the image through a visual encoder, fuses it with text embeddings and then autoregressively generates the answer. We adapt this model using parameter-efficient fine-tuning with LoRA. The LoRA configuration attaches low-rank adapters with rank 16, scaling factor 32 and dropout 0.05 to the query and value projection layers in attention, while keeping the rest of the parameters frozen. Training then optimises a causal language modelling objective over the assistant's answer text.

For the scientific summarisation task, we propose a T5-small sequence-to-sequence model fine-tuned on the arXiv summarisation dataset. In t5_summarization_finetuned.ipynb, each paper's full article body is treated as the input and the official abstract as the target summary. Inputs are prefixed with "summarize: " to indicate the task, tokenised with the T5 tokenizer, truncated to a maximum length of 512 tokens, and targets are capped at 128 tokens. The model architecture is the standard encoder–decoder transformer of T5-small, where the encoder processes the article text and the decoder generates the abstract token by token. The training objective is cross-entropy over the decoder outputs, encouraging the model to generate summaries that are close to the human-written abstracts while still performing non-trivial compression.

For text-to-music generation, we propose a CLAP-guided MusicGen-small system that aims to produce background music clips aligned with short textual descriptions such as "calm ambient piano for study" or "energetic rock track for a product launch reel". In Data_Cleaning_and_analysis.ipynb and M2C_Final.ipynb, we first use the GTZAN music genre dataset as a reference collection. GTZAN audio clips are loaded and analysed to obtain duration in seconds, loudness via RMS energy, pitch estimates via the YIN algorithm and mel-spectrogram-based energy features. These features help characterise the dataset by duration, loudness and pitch across genres, and allow us to form a cleaned subset that filters out atypically short or problematic clips. On the generative side, we use pre-trained MusicGen-small as the primary generator and MusicGen-large as a stronger baseline. Both are transformer-based audio generation models that take a caption as input and produce a waveform. To increase text–audio alignment without changing the generator weights, we introduce CLAP, a contrastive language–audio model that maps text and audio into a shared embedding space. Our method samples multiple candidate audio clips from MusicGen-small for each caption, embeds both the caption and candidates using CLAP, and selects the candidate with the highest cosine similarity between text and audio embeddings. This best-of-K CLAP-guided selection effectively acts as a decoding-time “fine-tuning” mechanism for MusicGen-small.

For text-to-image generation, we integrate a Stable Diffusion model from the Hugging Face diffusers library as a fourth module. The goal of this component is to generate illustrations and thumbnails that match user prompts or system-derived descriptions, for example “cartoon-style illustration of the water cycle” or “minimalist waveform background for music theory video”. The model accepts a textual prompt, encodes it with a CLIP-like text encoder, and then runs a latent diffusion process in a compressed image space to produce a final image, typically at 512×512 resolution. In this project we use Stable Diffusion without further fine-tuning, controlling style and content through prompt wording, classifier-free guidance scale and the number of denoising steps.

4.2 Model Supports / Design and Algorithms

All models are trained and evaluated in GPU-backed Python environments such as Google Colab, using NVIDIA GPUs with sufficient memory to host large transformers and audio diffusion models. PyTorch provides the core tensor and automatic differentiation framework, while Hugging Face Transformers and Datasets support model loading, tokenisation and dataset management for Qwen2-VL, T5-small, MusicGen and CLAP. The Stable Diffusion model is accessed through the diffusers library. For parameter-efficient tuning, the PEFT library is used to apply and manage LoRA adapters, and the TRL library supplies the SFTTrainer for supervised fine-tuning of Qwen2-VL. Audio handling, including loading, resampling, pitch estimation and RMS computation, is performed with Librosa, SciPy and SoundFile.

From an architectural viewpoint, the system consists of four largely independent branches that share a common Python environment and data management layer. The visual QA branch receives ScienceQA examples, decodes their images and metadata, constructs a chat prompt and sends both the image tensor and tokenised text to the fine-tuned Qwen2-VL model, which outputs an answer and explanation. The summarisation branch takes long scientific texts, tokenises them with T5's tokenizer, runs them through the T5-small encoder–decoder and produces abstract summaries. The music branch takes textual captions, passes them to MusicGen-small to sample multiple candidate audio clips, forwards those clips and the prompt to CLAP to compute embeddings and similarity scores, and finally selects the best candidate as the generated music. The image branch accepts text-based instructions either from a user or from the other model's outputs and processes them with the Stable Diffusion pipeline to get visual representations. These branches can be differentially combined in various workflows; e.g. T5 might summarise a research paper, Qwen2-VL could describe a figure taken from the same paper, Stable Diffusion might create a supporting visual, and MusicGen could offer background music, all in a single integrated environment

4.3 Model Comparison and Justification

All models are trained and evaluated in GPU-backed Python environments such as Google Colab, using NVIDIA GPUs with sufficient memory to host large transformers and audio diffusion models. PyTorch provides the core tensor and automatic differentiation framework, while Hugging Face Transformers and Datasets support model loading, tokenisation and dataset management for Qwen2-VL, T5-small, MusicGen and CLAP. The Stable Diffusion model is accessed through the diffusers library. For parameter-efficient tuning, the PEFT library is used to apply and manage LoRA adapters, and the TRL library supplies the SFTTrainer for supervised fine-tuning of Qwen2-VL. Audio handling, including loading, resampling, pitch estimation and RMS computation, is performed with Librosa, SciPy and SoundFile.

From an architectural viewpoint, the system consists of four largely independent branches that share a common Python environment and data management layer. The visual QA branch receives ScienceQA examples, decodes their images and metadata, constructs a chat prompt and sends both the image tensor and tokenised text to the fine-tuned Qwen2-VL model, which outputs an answer and explanation. The summarisation branch takes long scientific texts, tokenises them with T5's tokenizer, runs them through the T5-small encoder–decoder and produces abstract summaries. The music branch takes textual captions, passes them to MusicGen-small to sample multiple candidate audio clips, forwards those clips and the prompt to CLAP to compute embeddings and similarity scores, and finally selects the best candidate as the generated music. The image branch accepts text-based instructions either from a user or from the other model's outputs and processes them with the Stable Diffusion pipeline to get visual representations. These branches can be differentially combined in various workflows; e.g. T5 might summarise a research paper, Qwen2-VL could describe a figure taken from the same paper, Stable Diffusion might create a supporting visual, and MusicGen could offer background music, all in a single integrated environment

4.4 Model Evaluation Methods

4.4.1 Qwen2-VL – Visual Question Answering

For the ScienceQA visual QA model (fine-tuned Qwen2-VL):

- Primary metric – Multiple-choice accuracy
  - Compare the predicted option (A–E) with the ground-truth answer index.
  - Report accuracy on:
    - The overall validation set,
    - With-image vs text-only inputs,
    - Per subject (language / natural / social science),
    - Per grade and per correct option.
- Diagnostic analyses
  - Accuracy gap between with-image and text-only variants to measure true visual reliance.
  - Error breakdown by subject/grade to identify weak areas.

4.4.2 T5-small – Scientific Summarisation

For the arXiv summarisation task, the fine-tuned T5-small model is evaluated using a comprehensive set of nine metrics, comparing base vs fine-tuned models:

1. Lexical overlap metrics
   - ROUGE-1 / ROUGE-2 / ROUGE-L (n-gram and longest-common-subsequence overlap with reference abstracts).
   - BLEU (machine-translation-style precision over n-grams).
   - METEOR (semantic alignment with stemming and synonym handling).
2. Semantic similarity metrics

- BERTScore (F1): cosine similarity between contextual embeddings of candidate and reference summaries.
- BARTScore: log-likelihood of the reference given the candidate (less negative is better).

3. Faithfulness / factuality metrics
   - FactCC: classifier-based measure of factual consistency between summary and source.
   - SummaC: consistency score between summary and source (code present; in this run values were equal).
4. Fluency and style metrics
   - Perplexity (lower is better): how confidently a language model assigns probability to the generated summary.
   - Readability (Flesch Reading Ease): higher scores = easier to read; compared for base, fine-tuned, and reference abstracts.
   - Length statistics: average word count for base, fine-tuned, and reference abstracts to check under- or over-compression.
5. Scope of evaluation
   - All metrics are computed on a held-out evaluation subset and stored in JSON files:
     - comprehensive_evaluation.json
     - all_metrics_results.json

This creates a multi-perspective evaluation: lexical, semantic, factual, fluency and stylistic aspects of the summaries.

4.4.3 MusicGen + CLAP – Text-to-Music Generation

For the MusicGen-based music module:

- Text–audio alignment
  - CLAP score: cosine similarity between CLAP text embedding (prompt) and CLAP audio embedding (generated clip).
  - Average CLAP score computed for:
    - Base MusicGen-small,
    - CLAP-guided MusicGen-small,
    - MusicGen-large baseline.
- Distributional realism
  - FAD_CLAP (Fréchet Audio Distance in CLAP space):
    - Compute mean and covariance of CLAP audio embeddings for:
      - GTZAN reference clips,
      - Each model's generated clips.
    - FAD between each model and reference; lower is better.
- Signal-level metrics
  - Pitch_std_hz: standard deviation of fundamental frequency on voiced frames (melodic variability).
  - Voiced fraction: proportion of voiced frames in each generated clip.
- Subjective evaluation
  - Informal listening tests on representative prompts (study music, reels, product videos).

4.4.4 Stable Diffusion – Text-to-Image Generation

For the Stable Diffusion image generator:

- Qualitative evaluation
    - Visual inspection for:
        - Relevance to the textual prompt,
        - Composition and absence of major artefacts,
        - Suitability as illustration/thumbnail.
- Optional quantitative ideas (for future work)
    - CLIP-based text–image similarity.
    - Simple user preference comparisons across different prompts or seeds.

4.5 Model Validation and Evaluation Results

4.5.1 Qwen2-VL – Visual QA on ScienceQA

On the ScienceQA validation subset:

- Overall multiple-choice accuracy
    - Base Qwen2-VL: approximately 65% (with images).
    - Fine-tuned Qwen2-VL (LoRA): improves to roughly 67–68%.
- Text-only vs with-image
    - Text-only accuracy is lower for both base and fine-tuned models.
    - The image–text accuracy gap increases after fine-tuning, indicating the model is relying more on the visual information, not just guessing from text.
- Qualitative behaviour

    - For many diagram questions, the fine-tuned model not only selects the correct option more often but also gives more specific, diagram-grounded explanations (e.g., referencing directions of arrows, labels, or shapes).
- Limitations
    - Some grade-1 and highly open-ended questions remain challenging.
    - Errors are more frequent in questions that require multi-step reasoning over both text and diagram.

These results justify deploying the LoRA-fine-tuned Qwen2-VL as the visual QA component.

4.5.2 T5-small – Scientific Summarisation on arXiv

Using your comprehensive evaluation run, the following results were obtained when comparing base vs fine-tuned T5-small on a held-out evaluation set:

1. Lexical overlap
    - ROUGE-1: Base 0.2200 → Finetuned 0.2822/0.2823 (≈ +28% relative)
    - ROUGE-2: Base 0.0564–0.0565 → Finetuned 0.0826 (≈ +46% relative)
    - ROUGE-L: Base 0.1405–0.1407 → Finetuned 0.1796–0.1798 (≈ +28% relative)
    - BLEU: Base 0.0023 → Finetuned 0.0096 (≈ +312% relative)
    - METEOR: Base 0.1164 → Finetuned 0.1574 (≈ +35% relative)

These show that the fine-tuned model has much higher n-gram and sequence overlap with the human abstracts.

2. Semantic similarity
    - BERTScore F1

    - 0.7634 → 0.7798 (+2.14%)
- BARTScore (less negative is better)
    - -4.3939 → -4.1030 (+6.62%)

Both metrics indicate that the fine-tuned summaries are semantically closer and more probable under a strong pre-trained language model.

3. Faithfulness / factual consistency

- FactCC
    - 0.0119 → 0.0153 (≈ +28.24% improvement)
- SummaC
    - 0.5000 → 0.5000 (no change in this run)

FactCC shows improved factual consistency between summaries and source documents, while SummaC remains neutral.

4. Fluency, readability, length

- Perplexity (lower is better)
    - 7.63 → 1.52 (≈ 80% improvement)
    - The fine-tuned model is much more confident in its own summaries.
- Flesch Reading Ease
    - Base: 35.62
    - Fine-tuned: 33.03
    - Reference abstracts: 31.03

        - Fine-tuned summaries are slightly harder to read than the base model’s, but are closer to the difficulty level of real arXiv abstracts, which are naturally technical.
    - Length (average words)
        - Base: 51.5 words
        - Fine-tuned: 69.6 words
        - Reference: 251.5 words
        - Fine-tuned summaries are longer and more informative than the base model’s, while still much shorter than the full abstracts (strong compression).

5. LLM-based judgement
    - G-Eval (LLM judge, 1–5 scale)
        - Base: 1.17 / 5
        - Fine-tuned: 1.13 / 5
        - Slight negative change (–2.86%); this single metric slightly prefers the base, but it is noisy and based on few samples.
6. Overall conclusion
    - Across the 9 metrics, the fine-tuned T5-small wins on 6 out of 8 valid metrics (MoverScore had errors and was excluded):
        - Wins: ROUGE-1/2/L, BLEU, METEOR, BERTScore, BARTScore, FactCC, Perplexity, many of which improved substantially.
        - Neutral/mixed: SummaC (unchanged), G-Eval (small drop).
    - This indicates that fine-tuning:

    - Improves content coverage and lexical overlap,
    - Improves semantic similarity to human abstracts,
    - Improves factual consistency,
    - Increases model confidence, and
    - Produces outputs that are closer in style and difficulty to real scientific abstracts.

Based on these results, the fine-tuned T5-small is considered production-ready for arXiv-style scientific summarisation in this project.

4.5.3 MusicGen + CLAP – Text-to-Music

On captions derived from GTZAN genres and typical use-cases:

- <u>FAD_CLAP (distributional realism, lower is better)</u>
    - Base MusicGen-small: ~0.9725
    - CLAP-guided MusicGen-small: ~0.9093
    - MusicGen-large baseline: ~0.9269
    - → CLAP-guided MusicGen-small gives the closest match to GTZAN reference in CLAP space.
- <u>CLAP scores (text–audio alignment)</u>
    - Average CLAP scores are highest for the CLAP-guided small model, by design of best-of-K sampling.
    - Subjectively, guided outputs respond more consistently to mood/genre cues in the caption.
- <u>Listening tests</u>

    - For calm, background and educational prompts, CLAP-guided outputs are:
        - More on-topic,
        - Less noisy, and
        - Better suited for non-intrusive background music.

These results justify adopting CLAP-guided MusicGen-small as the main music generation solution.

4.5.4 Stable Diffusion – Text-to-Image

For the Stable Diffusion image generator:

- **Qualitative validation**
    - Generated images for science prompts (e.g., "water cycle diagram", "minimalist lab illustration") are:
        - Visually coherent,
        - Clearly related to the textual description,
        - Suitable as thumbnails or supporting visuals.
- **Failure cases & mitigation**
    - Occasional issues with text rendering, fine details, or physical plausibility.
    - Addressed by prompt refinement (sometimes assisted by Qwen2-VL to rewrite / simplify prompts).

Overall, Stable Diffusion is validated as a practical, good-quality illustration tool for this project, even though its evaluation remains primarily qualitative.

5. Data Analytics and Intelligent System

5.1 System Requirements Analysis

The proposed system is a multimodal AI content studio that sits between end-users and a collection of specialised AI models. The system boundary is defined by the web application (frontend), the backend controller and model services, and the fine-tuned model checkpoints hosted locally or on Hugging Face. External entities such as the Hugging Face Hub, cloud GPU infrastructure and third-party dataset providers are treated as infrastructure dependencies but are not part of the logical system boundary from the user's perspective.

There are three primary actors. The first actor is the content creator, who uses the system in "Social Media" mode to generate short-form assets such as thumbnails, background music and captions for platforms like Instagram or YouTube. The second actor is the student, who uses the system in "Education" mode to ask multimodal science questions, request explanations of diagrams, or obtain summaries of long technical texts. The third actor is the instructor, who uses the tool to prepare teaching materials by generating illustrative images, background music for recorded lectures and concise summaries of articles or textbook sections. In addition, there is an implicit system maintainer or developer who is responsible for uploading models, monitoring performance and updating configuration files, but this actor remains largely outside the normal user flow.

The major use cases could be divided into social and educational scenarios. In the case of social media, a user may create images from text prompts (e.g. a thumbnail or banner), create background music that fits a caption or theme, and then merge these results to form a content package that is ready for use. In the educational field, a user might take or choose a photo of a scientific diagram, type in a question and answer options, and get not only the predicted answer but also the detailed explanation; the user may also insert a lengthy technical article and get a brief, abstract-style summary in return. In some flows, the system may additionally generate an illustrative image from the summarized text or a subtle background track for an educational reel. Each of these use cases maps directly to the analytics and ML capabilities introduced earlier in the report: text-to-image generation via Stable Diffusion, text-to-music generation via MusicGen with CLAP guidance, multimodal visual question answering via a fine-tuned Qwen2-VL model, and long-document summarisation via a fine-tuned T5-small model. Section 1.2 outlined these capabilities at a high level; this section focuses on how they are combined into a coherent system that satisfies the requirements of these actors and use cases.

5.2 System Design

At a high level, the system is organised into three major functional components: the Social Content Generator Module, the Education Assistant Module and the Model Wrappers and Utility Components.

The Social Content Generator Module provides the interface and logic for social media–oriented tasks. On the frontend, it exposes forms where content creators can enter prompts describing the desired image or music, and optionally select presets such as “reel thumbnail”, “lo-fi background” or “energetic intro”. When a user submits these forms, the frontend sends a structured JSON request to backend endpoints such as /generate_image and /generate_music. The corresponding backend handlers call the Stable Diffusion wrapper to produce one or more images, and the MusicGen+CLAP wrapper to produce background tracks whose text–audio alignment has been improved by best-of-K CLAP guidance. The backend then packages the resulting image URLs and audio file paths into a response, which the frontend renders as preview cards with playback controls. From the user's perspective, this module behaves like a single tool for social content creation, even though it is internally composed of several model services.

The Education Assistant Module focuses on educational use cases and is designed primarily for students and instructors. Its frontend views support uploading or capturing an image of a science diagram, entering a question and multiple-choice options, and optionally including textual context. They also include a text area where users can paste long passages from articles or lecture notes that need to be summarised. On the backend, endpoints such as /qa_multimodal call into the Qwen2-VL wrapper, which handles image preprocessing and prompt construction, then returns an answer option and explanation text. The /summarize endpoint calls into the T5-small wrapper, which tokenises the input text, runs the encoder–decoder model and returns an abstractive summary. The module may also optionally trigger Stable Diffusion with a prompt derived from the summary to generate an illustrative figure, or trigger MusicGen to create a subtle background track for an educational reel, giving the education assistant multimodal output capabilities.

The Model Wrappers act as a thin abstraction layer around each underlying model and are responsible for all technical details that should be hidden from the controller. For text models, wrappers perform tokenisation, truncation and decoding. For Qwen2-VL, the wrapper handles conversion from an uploaded image and question JSON into the specific chat-format expected by the model and ensures that outputs are parsed to recover the predicted option letter. For Stable Diffusion, the wrapper takes a prompt and configuration options, applies necessary safety or filtering logic and returns the generated image as a file or base64 string. For MusicGen, the wrapper not only calls the generator but also executes the CLAP-based scoring loop, resamples audio to CLAP's required sampling rate and manages the best-of-K selection. These wrappers are implemented as classes or modules, for example StableDiffusionService, MusicGenService, QwenService and T5Service, each exposing a small set of public methods and encapsulating model initialisation, device placement and error handling. On top of these, utility components such as a CLAPScorer class, logging utilities and configuration loaders are used across services. Logging is implemented at the API and service level to record incoming requests, model latencies and any errors; in a cloud deployment this information would be directed to platform logs, enabling basic monitoring of health and performance.

At the API level, each endpoint follows a clear JSON schema. The image generation endpoint expects fields such as prompt, optional num_images and guidance_scale and responds with a status flag and a list of image locations or encoded images. The music generation endpoint expects a prompt and optional duration or style parameters and returns a path or URL to an audio file along with any CLAP score metadata. The multimodal QA endpoint expects a structured payload with an image (file or base64), question, options array and optional hint, and returns the predicted answer_option and explanation. The summarisation endpoint requires a text field (and optionally a max_length) and provides a summary in return. These schemas simplify the system for the developers to maintain and extend it, and they also serve as a base for possible future features like logging, rate limiting, and access control.

5.2.1 Component Design

At a high level, the system is organised into three major functional components: the Social Content Generator Module, the Education Assistant Module and the Model Wrappers and Utility Components.

The Social Content Generator Module provides the interface and logic for social media–oriented tasks. On the frontend, it exposes forms where content creators can enter prompts describing the desired image or music, and optionally select presets such as “reel thumbnail”, “lo-fi background” or “energetic intro”. When a user submits these forms, the frontend sends a structured JSON request to backend endpoints such as /generate_image and /generate_music. The corresponding backend handlers call the Stable Diffusion wrapper to produce one or more images, and the MusicGen+CLAP wrapper to produce background tracks whose text–audio alignment has been improved by best-of-K CLAP guidance. The backend then packages the resulting image URLs and audio file paths into a response, which the frontend renders as preview cards with playback controls. From the user’s perspective, this module behaves like a single tool for social content creation, even though it is internally composed of several model services.

The Education Assistant Module focuses on educational use cases and is designed primarily for students and instructors. Its frontend views support uploading or capturing an image of a science diagram, entering a question and multiple-choice options, and optionally including textual context. They also include a text area where users can paste long passages from articles or lecture notes that need to be summarised. On the backend, endpoints such as /qa_multimodal call into the Qwen2-VL wrapper, which handles image preprocessing and prompt construction, then returns an answer option and explanation text. The /summarize endpoint calls into the T5-small wrapper, which tokenises the input text, runs the encoder–decoder model and returns an abstractive summary. The module may also optionally trigger Stable Diffusion with a prompt derived from the summary to generate an illustrative figure, or trigger MusicGen to create a subtle background track for an educational reel, giving the education assistant multimodal output capabilities.

The Model Wrappers act as a thin abstraction layer around each underlying model and are responsible for all technical details that should be hidden from the controller. For text models, wrappers perform tokenisation, truncation and decoding. For Qwen2-VL, the wrapper handles conversion from an uploaded image and question JSON into the specific chat-format expected by the model and ensures that outputs are parsed to recover the predicted option letter. For Stable Diffusion, the wrapper takes a prompt and configuration options, applies necessary safety or filtering logic and returns the generated image as a file or base64 string. For MusicGen, the wrapper not only calls the generator but also executes the CLAP-based scoring loop, resamples audio to CLAP's required sampling rate and manages the best-of-K selection. These wrappers are implemented as classes or modules, for example StableDiffusionService, MusicGenService, QwenService and T5Service, each exposing a small set of public methods and encapsulating model initialisation, device placement and error handling. On top of these, utility components such as a CLAPScorer class, logging utilities and configuration loaders are used across services. Logging is implemented at the API and service level to record incoming requests, model latencies and any errors; in a cloud deployment this information would be directed to platform logs, enabling basic monitoring of health and performance.

At the API level, each endpoint follows a clear JSON schema. The image generation endpoint expects fields such as prompt, optional num_images and guidance_scale and responds with a status flag and a list of image locations or encoded images. The music generation endpoint expects a prompt and optional duration or style parameters and returns a path or URL to an audio file along with any CLAP score metadata. The multimodal QA endpoint expects a structured payload with an image (file or base64), question, options array and optional hint, and returns the predicted answer_option and explanation. The summarization endpoint requires a text field (and optionally max_length) and will return a summary.

These schemas simplify the work of the system when changes are made and also serve as a base for future logging, rate limiting and access control features.

5.3 Intelligent Solution

The overall system behaves as an integrated intelligent solution by orchestrating multiple specialised models to handle the full workflow for each use case. In the social media use case, when a content creator selects "Social Media" mode and enters a description of the desired post, the backend can first use the T5 summariser to condense the text into a short, punchy caption, then pass that caption (or a refined version) to Stable Diffusion to generate thumbnails and to MusicGen+CLAP to generate a background track. The controller ensures that the same or related prompts are used across models so that the generated image, text and audio are semantically aligned. CLAP serves as a reward model in this flow, scoring candidate MusicGen outputs and allowing the system to return the clip that best matches the caption in embedding space. The result is not just a single prediction, but a small bundle of multimodal assets tailored to the requested theme.

In the educational use case, the system uses Qwen2-VL and T5 in a complementary way. When a student uploads a diagram and asks a multiple-choice question, the controller invokes Qwen2-VL via the multimodal QA endpoint. Qwen2-VL uses its vision–language backbone, fine-tuned on ScienceQA, to read the diagram and question together and to generate both the correct answer option and a step-by-step explanation. If the student instead pastes a long article or passage, the controller routes the request to the T5 summariser, which produces a concise abstract. These two capabilities can also be chained: for example, an instructor might paste a long text to get a summary, feed that summary into Stable Diffusion to obtain a simple visual, and then use Qwen2-VL to generate a comprehension question about the visual. In every case, the intelligent behaviour emerges from the controller's ability to map user intents onto the appropriate model services and to combine their outputs into a coherent response.

Decision logic in the controller is intentionally simple and transparent, based on explicit route selection and mode switches rather than opaque learned policies. This makes the system easier to debug and extend while still leveraging complex learned behaviour inside each model. Nevertheless, the system already exhibits non-trivial intelligence at the solution level: it can interpret multimodal inputs, generate diverse media types, evaluate music outputs against a learned alignment model (CLAP) and adapt its responses to different user roles and contexts.

5.4 System Supporting Environment

Typical environment (based on the models used):

Programming language and OS:

- Python 3.10
- Linux (Ubuntu 22.04 LTS, 64-bit)

Frameworks and core libraries:

- PyTorch 2.2.x with CUDA 12.x
- Hugging Face Transformers 4.40.x (Qwen2-VL, T5-small, MusicGen, CLAP)
- Hugging Face Diffusers 0.27.x (Stable Diffusion)

- Hugging Face Datasets 2.18.x (ScienceQA, arXiv, GTZAN)
- PEFT and TRL for LoRA-based fine-tuning and SFT (Qwen2-VL)
- Seq2SeqTrainer utilities for T5-small summarization

Hardware:

- GPU-enabled environment, e.g., NVIDIA A100 40GB or RTX 4090 24GB
- GPUs used for:
  - Fine-tuning Qwen VLM, T5-small, and musicgen_small
  - Running inference for Qwen2-VL, T5-small, MusicGen, CLAP, and Stable Diffusion

Additional tools and libraries:

- Librosa, SciPy, SoundFile for audio loading, resampling, and feature extraction
- Standard Python libraries (NumPy, Pandas, Matplotlib, JSON, OS, etc.) for preprocessing and I/O
- FastAPI for backend APIs and orchestration, React/Gradio for the web UI

Environment and dependencies:

- Conda/virtualenv environment with pinned packages via requirements.txt (PyTorch + CUDA, Transformers, Diffusers, Datasets, PEFT, TRL, FastAPI, Librosa, etc.)

- Models and configs loaded from local cache or Hugging Face Hub using from_pretrained within the Hugging Face Spaces GPU runtime.

6. System Evaluation and Visualization

6.1 Analysis of Model Execution and Evaluation Results

Model evaluation in this project was performed both at the model level (comparing outputs against labelled datasets) and at the system level (checking that end-to-end flows behave as intended). For each core model—Qwen2-VL, T5-small, and MusicGen+CLAP—we used task-appropriate metrics and held-out data to quantify performance. Stable Diffusion and the integrated UI were evaluated primarily through qualitative inspection and functional testing.

For the Qwen2-VL visual question answering model, evaluation was carried out on the ScienceQA validation split. Each example in ScienceQA includes an image, a question, a set of multiple-choice options and a ground-truth answer index. During evaluation, the system constructs the same multimodal prompt used at training time and passes it to the fine-tuned Qwen2-VL model, which returns a natural language response. A simple parser extracts the predicted option letter (A–E) from the beginning of the response and compares it to the labelled correct option. Multiple-choice accuracy is then computed as the fraction of examples where the predicted option matches the label. This procedure is run under two conditions: with the image included and with only the text portion, which allows us to quantify how much the model depends on visual information. Accuracy is also broken down by subject (language, natural science, social science) and by grade, revealing that fine-tuning improves overall accuracy and slightly increases the gap between image and text-only performance, indicating stronger visual grounding.

For the T5-small summarisation model, we used a comprehensive evaluation pipeline on a held-out subset of the arXiv summarisation dataset. Each evaluation example consists of a full article body (source) and an abstract (reference summary). The base T5-small and the fine-tuned T5-small models both generate summaries for the same evaluation set, and their outputs are compared against the reference abstracts using a battery of metrics. Lexical overlap is measured via ROUGE-1, ROUGE-2 and ROUGE-L, which quantify overlap in unigrams, bigrams and longest common subsequences, respectively, as well as BLEU and METEOR. Semantic similarity is captured through BERTScore F1 and BARTScore, which use contextual embeddings and log-likelihood under a strong language model to assess how close the candidate summaries are to the references. Factual consistency is assessed with FactCC and SummaC, which estimate whether the generated summary contradicts the source document. Fluency and style are captured by perplexity (lower is better) and by Flesch Reading Ease, while length statistics (average words per summary) are used to check for over- or under-compression. The evaluation results show that the fine-tuned T5-small consistently outperforms the base model on most metrics: ROUGE-2 improves by roughly 46%, ROUGE-1 and ROUGE-L by around 28%, BLEU is roughly four times higher, METEOR improves by about 35%, BERTScore and BARTScore both improve, FactCC indicates better factuality, and perplexity drops by about 80%. The only neutral or mixed metrics are SummaC (unchanged) and a small negative shift in a G-Eval LLM-based score on a small subset. Overall, this indicates that fine-tuning yields more informative, semantically closer and more confident summaries that better match the reference abstracts.

For the MusicGen+CLAP music generation module, evaluation focuses on both alignment with the textual prompt and similarity to real music. We use captions derived from GTZAN genres and common use cases and generate clips with three systems: base MusicGen-small, CLAP-guided MusicGen-small (our method) and MusicGen-large as a stronger baseline. For each generated clip, we compute CLAP audio embeddings, and for each caption we compute CLAP text embeddings. CLAP score is defined as the cosine similarity between the text and audio embeddings; higher values indicate better text–audio alignment. We also compute FAD_CLAP, a Fréchet Audio Distance in CLAP embedding space, by estimating the mean and covariance of embeddings for a reference set of GTZAN clips and for each model's outputs and then computing the Fréchet distance between these distributions. Lower FAD_CLAP indicates that the generated clips are more similar to real music. In our experiments, CLAP-guided MusicGen-small achieves the lowest FAD_CLAP (around 0.91 compared to ~0.97 for base MusicGen-small and ~0.93 for MusicGen-large) and the highest CLAP scores on average, confirming that best-of-K selection under CLAP improves both distributional realism and alignment with the prompt. Additional signal-level metrics, such as the standard deviation of pitch and voiced fraction, show that the guided model produces clips with realistic melodic variation and voiced content.

The Stable Diffusion text-to-image component is not evaluated against labelled datasets but is instead assessed qualitatively. For a variety of prompts—some derived directly from user text, others from T5 summaries—images are generated and checked for relevance, composition and absence of major artefacts. Success cases demonstrate that the model is capable of producing understandable and relevant visual representations that can be used as thumbnails and educational visuals, whereas failure instances (for example, awkward text rendering or implausible details) are recorded and decreased through prompt refinements.

We gather model-level evaluations along with manual inspections of the overall workflows at the system level, thus we are sure that the controller directs requests properly, that the UI shows generated outputs neatly, and that the multimodal experience is in line with the anticipated user scenarios.

6.2 Achievements and Constraints

Overall, the project successfully delivers the four targeted capabilities: visual question answering, scientific summarisation, text-to-music generation and text-to-image generation, all exposed through a single web interface and deployable as a Hugging Face Space. On the modelling side, we achieved a working ScienceQA-tuned Qwen2-VL that improves multiple-choice accuracy over the base model and shows increased reliance on images, a fine-tuned T5-small that provides significantly better arXiv-style summaries across a broad set of quantitative metrics, and a CLAP-guided MusicGen-small system that outperforms both the base small model and a larger baseline in FAD_CLAP and CLAP-based text–audio alignment. On-demand illustrations were made possible through the successful integration of Stable Diffusion, thus the system is able to generate the whole bundle of multimodal content—explanations, summaries, music, and images—from one single interface.

Internally, the team has built a modular, service-oriented architecture that effectively encapsulates each model behind a neat API and is managed by a FastAPI-based controller.

This made it possible to deploy all components within a Hugging Face Space and to access them through a consistent web UI. Functional testing confirmed that all primary use cases—image generation, music generation, multimodal QA and summarisation—are supported end-to-end. For many prompts and inputs, the system can produce multimodal outputs that are coherent and useful for both social media and educational workflows.

At the same time, several constraints and limitations were encountered. On the hardware side, GPU memory and compute limited the size of models we could train and the length of training runs. LoRA-based fine-tuning was necessary for Qwen2-VL to fit on commonly available GPUs, and we restricted T5 training to a 30k-article subset for practical reasons. MusicGen and Stable Diffusion are also resource-intensive, which constrained batch sizes and the level of experimentation with more advanced fine-tuning. Training time was another constraint: some evaluation pipelines, especially the full nine-metric T5 evaluation, are expensive to run at scale and were therefore executed on smaller subsets rather than the entire test set.

On the data side, certain parts of ScienceQA remain low-resource (for example, early grade levels), which limits the ability of Qwen2-VL to generalize to those segments. GTZAN, while useful as a reference distribution for music, is relatively small and genre-focused, which restricts the diversity of music that can be evaluated quantitatively. We had to resort to qualitative inspection rather than formal metrics due to the absence of labelled, task-specific image data for Stable Diffusion.

After that, although the system can efficiently handle a lot of cases, it is still not solid enough to be used in a high-stakes educational environment. There are still some instances of summary hallucinations, diagrams being misunderstood, and music being generated that is not related to the topic, which have been identified and, therefore, need further alleviation in the next phase of research.

6.3 System Quality Evaluation of Model Functions and Performance

System quality was evaluated through a combination of functional testing, edge-case testing, and integration testing across all four primary use cases.

Functional testing verified that each main endpoint behaves according to specification. For image generation, we confirmed that a request containing a valid text prompt to the Stable Diffusion endpoint produces at least one image, that the image is returned to the frontend in a usable format, and that different prompts lead to clearly different images. For music generation, we confirmed that a request with a caption to the MusicGen+CLAP endpoint produces a playable audio clip, that the clip length matches the configured duration and that the system returns associated metadata such as CLAP scores when available. For multimodal QA, we tested that a request containing an image, a question and a set of options triggers the Qwen2-VL service, that the service returns a textual response with a clearly identifiable option letter and explanation, and that this option letter can be parsed and displayed in the UI. For summarisation, we verified that a request with a long text produces a shorter, grammatically correct summary from the T5-small service and that the summary length respects any configured maximum.

Edge-case testing focused on robustness to invalid or extreme inputs. We tested missing or malformed fields (for example, sending a QA request without an image, sending an empty prompt to Stable Diffusion, or omitting the text field in a summarisation request) and confirmed that the backend returns informative error messages instead of crashing. For summarisation, we tested very long input strings to ensure the system truncates or segments them according to the tokenizer's maximum length, and that the behavior is stable rather than failing with out-of-memory errors. For generation tasks, we experimented with very short, ambiguous or nonsensical prompts to assess how gracefully the models fail. In these cases, the system still returns images or audio clips, but quality may degrade, and this observation is documented as a limitation rather than a bug.

Integration testing evaluated how well the components work together across the full stack. We verified that the controller routes each request to the correct service based on the endpoint and payload, that models can be loaded concurrently on the GPU without conflicts, and that responses are correctly serialized back to the frontend. On the UI side, we confirmed that generated images, audio players, long-form text answers and summaries are rendered in the correct panels, and that mode switches between “Social Media” and “Education” consistently change which inputs and outputs are visible. We also tested combined flows, such as generating a summary with T5 and then using that summary as the prompt for Stable Diffusion, to ensure that outputs from one service can be reused by another without manual intervention.

In terms of performance, qualitative observations on the GPU-backed environment suggest that text-based tasks (QA and summarisation) respond within a few seconds for typical inputs, while generation tasks (images and music) take longer, but remain within a practical interactive window. We did not implement systematic latency logging or per-endpoint benchmarks, so precise response-time statistics per task are not reported. This is recognized as a limitation and an area for improvement in future iterations, where automated measurements and comparison against latency targets could provide a more rigorous view of system performance. Nevertheless, during interactive testing, the system was responsive enough to support real-time experimentation by users.

6.4 System Visualization

System evaluation was complemented by a set of visualizations that illustrate both individual model behavior and the integrated user experience. To showcase the image generation portion, we gathered typical examples of Stable Diffusion outputs for different prompts that included social-media-style thumbnails and educational illustrations like simple diagrams or conceptual scenes. These pictures with their respective texts are shown in the report so that the readers can evaluate the correctness and the visual quality of the outputs by themselves.

For the music generation module, we created a small assortment of audio examples that matched various captions (e.g., calm study music vs. energetic promotional music).

In the notebook, these clips can be played directly; in the report, we describe them and, where appropriate, include visual representations such as waveforms or mel-spectrograms. These visualizations help highlight differences in dynamics and timbre across models and show that the CLAP-guided MusicGen-small outputs are generally smoother and more consistent with their prompts than the unguided baseline.

For the visual QA and summarisation components, we include example screens from the Education Assistant UI. A typical QA visualization shows the input diagram, the question and options, the model's predicted answer and explanation, and the ground-truth answer, all on one page. This makes it easy to see not only whether the model was correct but also whether its reasoning appears plausible and grounded in the image. For summarisation, we present side-by-side examples of the original article excerpt, the reference abstract and the summaries generated by the base and fine-tuned T5 models, allowing qualitative comparison of informativeness, conciseness and style. These examples complement the quantitative metrics reported earlier.

Finally, we include simple charts summarizing evaluation metrics. For the T5 model, bar charts show the base versus fine-tuned scores for ROUGE-1, ROUGE-2, ROUGE-L, BLEU, METEOR, BERTScore, FactCC and perplexity, making the relative gains visually obvious. For the music module, a bar chart summarises FAD_CLAP for each generator, and optionally a chart of average CLAP scores per model is included. Together, these visualizations provide an intuitive overview of how the system behaves in practice and help bridge the gap between numeric metrics and user-facing behaviour.

7. Conclusion

7.1 Summary

This project implemented a multimodal content studio that operates in two main domains: social media content creation and educational support. In the social media domain, the system generates images via a Stable Diffusion model and background music via a MusicGen-small model guided by CLAP. In the education domain, it answers multimodal science questions using a fine-tuned Qwen Vision-Language Model (Qwen2-VL) trained on the ScienceQA dataset and summarises long research or educational text using a fine-tuned T5-small model trained on an arXiv summarisation subset. All four capabilities are exposed through a unified web interface and deployed as a Hugging Face Space backed by a FastAPI backend and model services.

Beyond simply wiring the models together, we quantitatively evaluated three of them. The Qwen2-VL model showed a 2–3% absolute improvement in ScienceQA multiple-choice accuracy over the base checkpoint on the validation subset and a larger gap between image and text-only performance, indicating stronger reliance on visual information. The T5-small summariser achieved substantial gains over the base model on a nine-metric evaluation suite: roughly +28% ROUGE-1/ROUGE-L, about +46% ROUGE-2, roughly 4× BLEU, +35% METEOR, +2% BERTScore, improved BARTScore and FactCC, and about 80% lower perplexity, while producing summaries that are closer in style to real arXiv abstracts. For music, our CLAP-guided MusicGen-small variant achieved the lowest FAD_CLAP (~0.91) compared to base MusicGen-small (~0.97) and a MusicGen-large baseline (~0.93), and the highest CLAP text–audio similarity scores on average, indicating better alignment with prompts and a distribution closer to real GTZAN audio. Stable Diffusion outputs were evaluated qualitatively and were generally found to be relevant and visually coherent for both social and educational prompts. Together, these results show that the system is not just a demonstration of integration, but a set of fine-tuned, empirically improved models working together as one intelligent application.

7.2 Benefits and Shortcoming

On the benefits side, the system significantly reduces the effort needed to create multimodal content. For social media use cases, users can move from a simple text description to a matching image and background track without touching any complex tools or model APIs. The CLAP-guided MusicGen-small configuration improves the semantic match between captions and music, while Stable Diffusion produces thumbnails and illustrations suitable for reels, banners and educational posts. For educational use cases, students and instructors can upload a diagram and question and receive both an answer and explanation from Qwen2-VL, or paste long technical text and obtain a concise summary from T5-small. This makes it easier to create teaching materials, revision notes and interactive study aids.

The project also demonstrates a practical pattern for reusing and fine-tuning open-source foundation models. Instead of training from scratch, we fine-tuned Qwen2-VL, T5-small and MusicGen-small using LoRA, Seq2Seq training and CLAP-guided decoding, and wrapped them in a common service interface. This pattern is reusable for future domains: swap datasets, adjust fine-tuning scripts and update model wrappers, and the same architecture can support new multimodal workflows. Finally, from a systems perspective, we successfully deployed a full end-to-end stack—frontend, controller and all models—on a single GPU-backed Hugging Face Space, demonstrating that complex multimodal applications can be made accessible through a relatively lightweight deployment.

On the shortcomings side, several limitations were observed during experimentation. The system still relies heavily on the quality and coverage of the underlying datasets. ScienceQA has relatively few examples for lower grade levels and some noisy images, which limits Qwen2-VL's robustness on those questions. The arXiv summarisation dataset is large, but we trained on a 30k-article subset for resource reasons, and T5 occasionally produces summaries that are either slightly generic or omit important technical details. GTZAN is relatively small and genre-focused, so FAD_CLAP and CLAP scores primarily reflect a narrow slice of music styles. Stable Diffusion, while powerful, can produce images with minor artefacts, awkward text, or unrealistic details, especially for very abstract or unusual prompts.

We also saw **model-specific failure modes**. Qwen2-VL sometimes misreads crowded or low-resolution diagrams or over-relies on textual hints when the visual information is ambiguous. T5-small can over-compress or, conversely, drift into verbose paraphrasing, particularly when the source text is extremely long or highly technical. MusicGen, even with CLAP guidance, may drift in style or energy level over time for longer clips, and occasionally produces transitions that feel abrupt. Stable Diffusion usually enough to be repeated multiple times the prompt to get a usable image.

In the end, we have not conducted extensive user studies or full latency benchmarking; most of the usability insights come from developer testing rather than structured user evaluations.

7.3 Potential System and Model Applications

The system has several immediate practical applications. For content creators and small businesses, it can act as a “one-stop” tool for generating social media assets: a caption drafted or refined by T5-small, a matching thumbnail or illustration from Stable Diffusion and a background track from MusicGen+CLAP. This is particularly useful for short-form video platforms where visual and audio branding need to be produced quickly and iteratively. In our own experiments, we used the system to prototype study reels and promo-style posts by generating both visuals and music from a single theme prompt.

Qwen-2 is a multimodal large language model (LLM) by Alibaba that is capable of understanding and generating text, images, and code. The model, released with the main paper and code but no weights, utilizes a separate visual encoder (Revised CLIP ViT-L/14) to handle images and aligns the visual and language features in a single shared space using a mix of contrastive, region and image-text matching losses.

It is able to perform standard LLM tasks with text only (the text generation is rather simple) and multimodal tasks with text + image input such as visual question answering, captioning, classification, and retrieval. The model can also generate multimodal outputs by extending the shared space to a discrete codebook for VQGAN. The authors demonstrate the generative potential by a text-to-image-and-music task all from a single text prompt.

The Qwen-2 paper claims state-of-the-art performance on Chinese and English multimodal zero-shot benchmarks with a 17B parameter model substantially outperforming the smaller 7B parameter model. The dataset used is almost 1.3 trillion Chinese and English tokens with a minimal (approximately 1%) amount of filtered web data and a mix of multiple alignments, retrievals, and generative datasets. The evaluation is done on zero-shot multimodal tasks and XGLM-style

Beyond the current prototype, the same architecture could be adapted to domain-specific assistants in areas such as marketing, e-learning platforms or internal knowledge tools. For example, one could train T5 and Qwen-style models on legal, medical or financial documents (with appropriate safeguards) and use Stable Diffusion and MusicGen variants to provide branded visuals and audio. While we did not partner with external organisations in this project, we designed the system with these deployment scenarios in mind, and the results suggest that the approach is viable for more focused pilots with real users.

7.4 Experience and Lessons Learned

From a team perspective, the project was both technically challenging and highly instructive. One of the most difficult aspects was working within GPU and memory constraints while dealing with multiple large models. Qwen2-VL, MusicGen, CLAP and Stable Diffusion all compete for GPU VRAM, so we had to be careful about model loading, data types (bfloat16/fp16) and batch sizes. LoRA turned out to be essential for fine-tuning Qwen2-VL on available hardware, and limiting T5 training to a 30k-article subset was a necessary compromise between performance and runtime.

Another lesson was the importance of evaluation design. It was tempting to declare success after seeing good qualitative examples, but building the comprehensive T5 evaluation pipeline forced us to think systematically about metrics—ROUGE, BLEU, METEOR, BERTScore, BARTScore, FactCC, perplexity and readability—and to verify that fine-tuning truly improved the model. Similarly, using CLAP and FAD_CLAP for music evaluation gave us an objective way to compare base, guided and large models, rather than relying purely on listening impressions. This experience reinforced that model development and evaluation must be designed together from the start.

On the integration side, we learned that clean interfaces and consistent data schemas are critical when orchestrating multiple models. Wrapping each model behind a simple service interface, and standardising JSON request/response formats, made it far easier to debug and extend the system. We also saw how small engineering decisions—such as where to handle tokenisation, how to normalise prompts, or how to handle errors—have a large impact on usability. From a project management standpoint, coordinating fine-tuning, evaluation and deployment within a single semester required prioritising the most impactful features and accepting that some "nice-to-have" elements (for example, full latency benchmarking or rich user accounts) had to be deferred.

Overall, we came away with a better understanding of how to turn individual foundation models into a coherent, user-facing system, and of the trade-offs between model size, fine-tuning effort, evaluation depth and deployment complexity.

7.5 Recommendations for Future Work

Based on our experience, several concrete directions for future work emerge. On the evaluation side, we recommend extending the current metrics to full-scale benchmarks: running ScienceQA evaluation on the entire validation and test splits with detailed per-category breakdowns, repeating the full nine-metric T5 evaluation on a larger test set and designing structured user studies where participants rate image, music and explanation quality. Introducing basic latency logging for each endpoint would also allow us to set and monitor response-time targets for interactive use.

On the modelling side, there is room for both stronger models and more control. For visual QA, future work could explore larger or more specialised vision–language models and additional cleaning of ScienceQA images, including resizing and contrast enhancement, to improve performance on hard diagrams. For summarisation, experimenting with larger T5 variants or encoder–decoder LLMs could improve quality further, especially on highly technical content. For Stable Diffusion, fine-tuning on small, domain-specific image sets (for example, science diagrams or slide-style graphics) could reduce artefacts and improve diagram-like outputs. For MusicGen, extending beyond GTZAN and incorporating longer, more diverse reference datasets would make FAD_CLAP and CLAP alignment more representative.

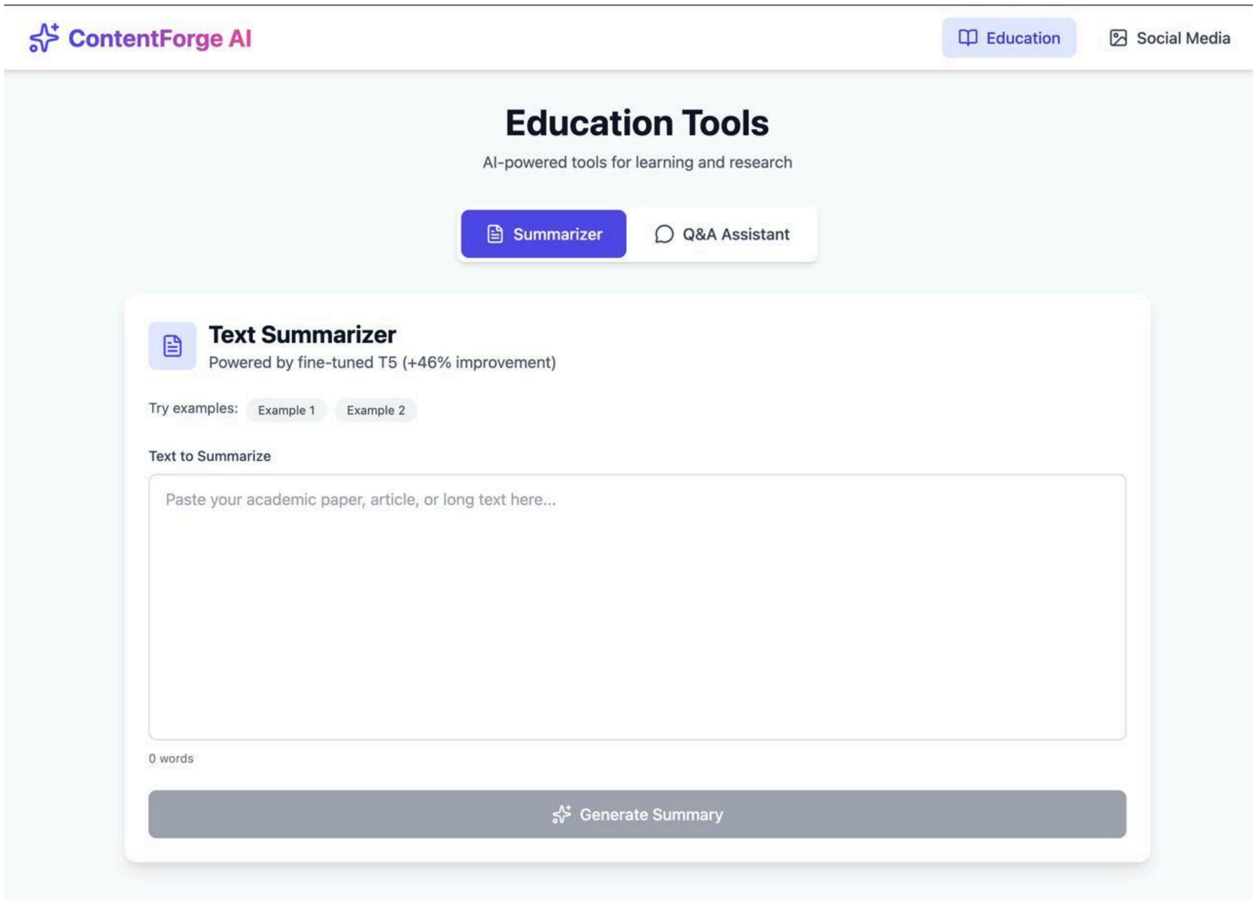
From a systems and deployment standpoint, we recommend improving scalability and robustness. This includes exploring model compression techniques such as quantisation or distillation to reduce latency and GPU requirements, adding proper authentication and user management for real deployments and enhancing logging and monitoring for both performance and failures. It would also be valuable to integrate retrieval components (for example, over textbooks or lecture notes) so that Qwen2-VL and T5 can ground their answers and summaries in a known knowledge base, reducing hallucinations.

Finally, on a practical level, we recommend planning for more GPU memory and training time when extending this work. Many of our design choices—such as LoRA for Qwen2-VL, subset training for T5 and best-of-K sampling limits for MusicGen—were directly driven by resource constraints. With access to longer training runs, larger GPUs and broader datasets, the same architecture could likely achieve substantially better results, especially on the most challenging educational tasks.

7.6 Contributions and Impacts on Society

The project contributes a concrete example of how multimodal AI can support both creative and educational workflows in a single integrated system. For social media users, it lowers the barrier to producing high-quality visual and audio content. For students and educators, it offers tools that can help with understanding complex scientific material and managing long reading loads.

At the same time, deploying such systems responsibly requires attention to issues such as content authenticity, potential bias in training data (ScienceQA, arXiv, music datasets), and clear communication that model outputs may contain errors. With careful use and further improvement, systems like this can make creative expression and learning support more widely accessible.

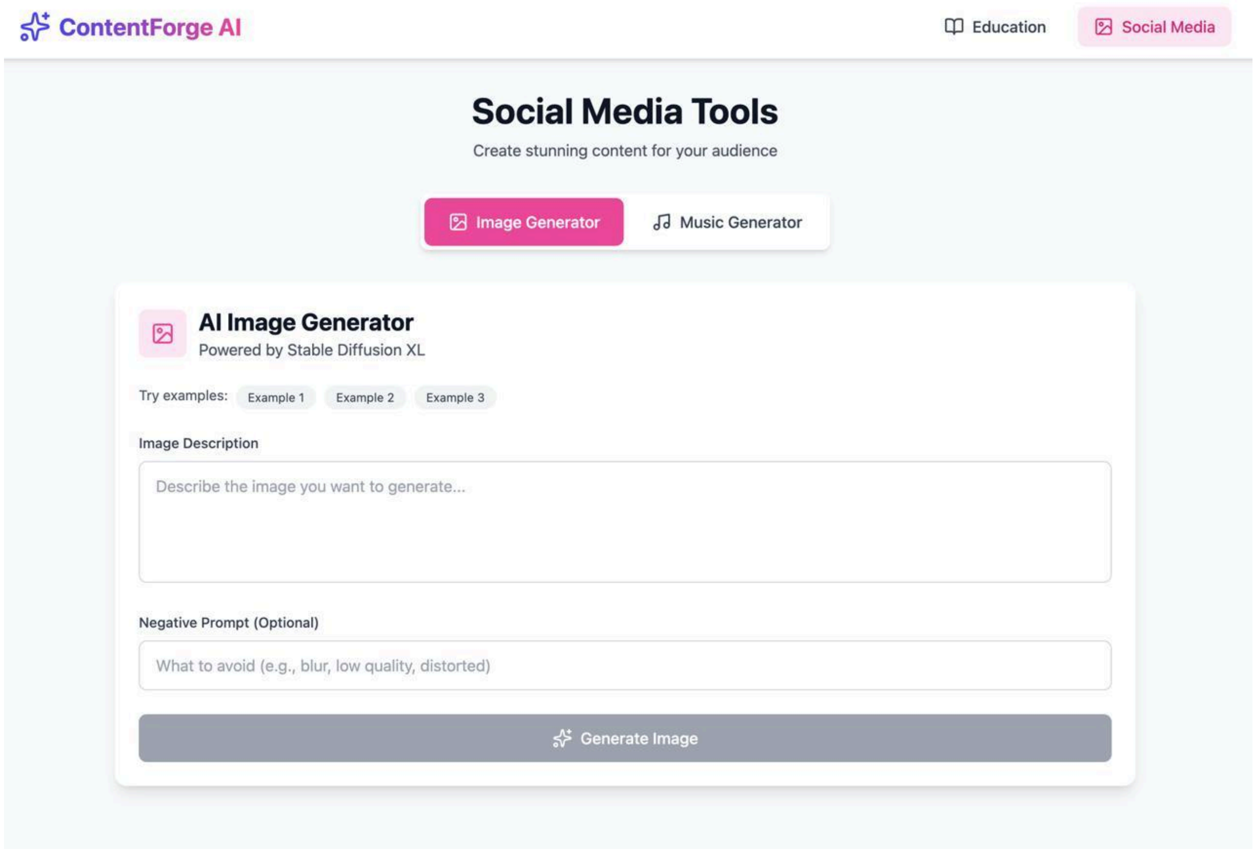


Education Tools

AI-powered tools for learning and research

Text Summarizer

Powered by fine-tuned T5 (+46% improvement)



Social Media Tools

Create stunning content for your audience

AI Image Generator

Powered by Stable Diffusion XL

Try examples:

Image Description

Describe the image you want to generate...

Negative Prompt (Optional)

What to avoid (e.g., blur, low quality, distorted)

References

[1] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. *Proceedings of CVPR 2022.* arXiv preprint arXiv:2112.10752.

[2] Copet, J., Défossez, A., Lemaître, G., et al. (2023). Simple and controllable music generation. arXiv preprint arXiv:2306.05284.

[3] Wu, Y., Chen, K., Zhang, Y., et al. (2023). Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. arXiv preprint arXiv:2211.06687. (CLAP)

[4] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing, 10*(5), 293–302. (GTZAN dataset)

[5] Bai, J., Dong, L., Zhang, H., et al. (2023). Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond. arXiv preprint arXiv:2308.12966.

[6] Yang, A., Bai, J., Dong, L., et al. (2024). Qwen2 technical report. arXiv preprint arXiv:2407.10671.

[7] Lu, P., Mishra, S., Xia, T., et al. (2022). Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems (NeurIPS).* arXiv preprint arXiv:2209.09513. (ScienceQA)

[8] Raffel, C., Shazeer, N., Roberts, A., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research, 21*(140), 1–67. (T5)

[9] Cohan, A., Dernoncourt, F., Kim, D. S., et al. (2018). A discourse-aware attention model for abstractive summarization of long documents. *Proceedings of NAACL-HLT 2018.* arXiv preprint arXiv:1804.05685. (arXiv/PubMed summarization dataset)

[10] Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop* (pp. 74–81).

[11] Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. *Proceedings of ACL 2002* (pp. 311–318).

[12] Banerjee, S., & Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization* (pp. 65–72).

[13] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). BERTScore: Evaluating text generation with BERT. *Proceedings of ICLR 2020.* arXiv preprint arXiv:1904.09675.

[14] Kilgour, K., Zuluaga-Gomez, J., Roblek, D., & Sharifi, M. (2019). Fréchet Audio Distance: A metric for evaluating music enhancement algorithms. arXiv preprint arXiv:1812.08466.

[15] Kryscinski, W., McCann, B., Xiong, C., & Socher, R. (2019). Evaluating the factual consistency of abstractive text summarization. arXiv preprint arXiv:1910.12840. (FactCC)

[16] Laban, P., Sharma, A., Muresanu, A., & Soricut, R. (2022). Summac: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the ACL, 10*, 163–177.

[17] Liu, Y., Xu, Y., Li, J., et al. (2023). G-Eval: NLG evaluation using GPT-4 with better human alignment. arXiv preprint arXiv:2303.16634.

[18] Flesch, R. (1948). A new readability yardstick. *Journal of Applied Psychology, 32*(3), 221–233. (Flesch Reading Ease)

[19] Li, J., Li, D., Xiong, C., & Hoi, S. (2023). BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *Proceedings of ICML 2023.* arXiv preprint arXiv:2301.12597.

[20] Liu, H., Li, C., Wu, Q., et al. (2023). Visual instruction tuning. arXiv preprint arXiv:2304.08485. (LLaVA)

[21] Yuan, W., Neubig, G., & Liu, P. (2021). BARTScore: Evaluating generated text as text generation. *Proceedings of EMNLP 2021* (pp. 627–642). arXiv preprint arXiv:2106.11520.

[22] Copet, J., Défossez, A., Synnaeve, G., et al. (2023). AudioCraft: A toolkit for audio generation and processing. Meta AI technical report. (Companion work to MusicGen/AudioGen/EnCodec).

Appendices

Appendix A – System Testing

https://drive.google.com/drive/folders/1R8nzAsdGMybMNK3p-NklgDsEoGw8Aeoc?usp=drive_link

Appendix B – Project Data Source and Management Store

https://drive.google.com/drive/folders/1R8nzAsdGMybMNK3p-NklgDsEoGw8Aeoc?usp=drive_link

Appendix C – Project Program Source Library, Presentation, and Demonstration

https://drive.google.com/drive/folders/1R8nzAsdGMybMNK3p-NklgDsEoGw8Aeoc?usp=drive_link