UCLouvain

epl

École polytechnique de Louvain

ArtEvoke: AI-powered platform to trigger memories in Alzheimer's patients using art imagery

Authors: **Augusto B. V. Silva, Marc Bejjani**
Supervisor: **Benoît Macq**
Readers: **Eric Piette, Tiffanie Godelaine, Sébastien Lugan**
Academic year 2024–2025
Master [120] in Computer Science and Engineering
Master [120] in Computer Science

Abstract

This thesis explores how artificial intelligence can support non-drug interventions for people living with Alzheimer's disease through the use of visual stimuli and art. We present a prototype platform that combines large multimodal models (LMMs), large language models (LLMs), and embedding-based vector search to connect personal stories with curated artworks. Designed to assist memory recall and cognitive engagement, ArtEvoke generates personalized visual narratives or retrieves relevant paintings based on the user's story. Art is used not only as a memory trigger but also as a tool to promote emotional connection and engagement. The prototype is intended for testing in future therapeutic and research settings, particularly as a scalable support tool for caregivers and clinical practitioners. We evaluate model performance, retrieval effectiveness using FAISS, and user interaction design with a focus on accessibility. These findings highlight the potential of AI tools to support scalable cognitive stimulation in dementia care.

Acknowledgements

These past six years, through both our bachelor's and master's studies, have been through a journey of personal and academic growth. During this time, we have learned more about ourselves and refined our academic and professional interests. This thesis marks the completion of our respective master's degrees: Augusto Silva in Computer Engineering, as part of a double degree with the University of São Paulo (USP), and Marc Bejjani in Computer Science.

We would like to express our gratitude to our supervisor, Prof. Benoît Macq, for offering us the opportunity to work on this project. We also thank Tiffanie Godelaine for her valuable feedback and support throughout both the writing process and the modeling of our platform. Our thanks also go to Sébastien Lugan for his technical assistance, and to Eric Piette for agreeing to be a reader.

Finally, we are deeply grateful to our friends and families for their constant support throughout this thesis and our academic journey. Your encouragement, patience, and understanding, especially during the more stressful moments, kept us going. Thank you for always being there.

Contents

Chapter 1

Introduction

1.1 Context

Alzheimer's disease (AD) is a common neurological disorder that affects the memory and behavior of the affected people. In fact, it is the leading cause of dementia, which is the severe loss of mental function, enough to greatly affect daily life negatively. It is estimated that around 55 million people currently suffer from this syndrome, and that number is expected to rise to nearly 150 million by the year 2050, based on estimates from the World Health Organization [1]. Consequently, as the global population ages due to a number of different factors, there is an increasing demand to innovate and develop accessible intervention methods for the mentioned disease.

While pharmaceutical treatments do exist today, they possess a limited effectiveness and scope especially due to the fact that they are not able to successfully tackle the progressive or degenerative nature of the disease. In light of this, other approaches are being explored, such as cognitive stimulation [2]. The latter is a field studied by both clinical and technical researchers, and is aimed at improving the quality of life of the patients by providing them with activities that stimulate their thinking, concentration, and other brain activities.

In this context, and as further discussed in the next Chapter (Section 2.1.4), recent research has explored the use of visual imagery as a cognitive stimulation tool for individuals with AD. Studies suggest that exposure to artworks can help trigger memory and activate broad neural pathways, offering therapeutic benefits. However, selecting appropriate images that resonate with a patient's experiences remains a non-trivial task. While expert input may enhance the relevance of selected stimuli, this manual process can be time-consuming and difficult to scale.

At the same time, the field of artificial intelligence (AI) has undergone rapid advancements, particularly in the development of models capable of interpreting and generating visual and textual information. These technologies can be leveraged to support and automate parts of the image selection and storytelling process, making such therapeutic interventions more accessible and less dependent on the time of specialists.

1.2 Objectives and Approach

The objective of this thesis is to develop ArtEvoke, a prototype application that leverages artificial intelligence and visual stimuli to provide an accessible tool for individuals with AD or other forms of dementia. By combining AI with curated artworks, ArtEvoke

aims to stimulate memory and cognitive engagement through personalized visual narratives. This work contributes to the field of non-pharmacological interventions for dementia by proposing a scalable platform that goes beyond expert-assisted cognitive stimulation. The prototype is intended to serve both as a research tool and, potentially, as a practical solution in future clinical applications.

To achieve this goal, the following approach was adopted:

- **Selection and evaluation of AI models:** Identify and benchmark the most suitable large multimodal models (LMMs), large language models (LLMs), and embedding strategies for description generation, storytelling, and similarity search.
- **Feature extraction from artworks:** Use AI-based tools to extract descriptive, emotional, and symbolic information from art datasets to enhance retrieval and interaction quality.
- **Design of interaction modes:** Develop and refine interaction modes that enable dynamic and engaging user experiences through visual content.
- **Prototype development:** Integrate all components into a functional and user-friendly prototype, with special attention to accessibility and cognitive usability for elderly users.

Figure 1.1 presents a simplified overview of the ArtEvoke platform. The process starts with two key inputs: **Feature Extraction**, where features are extracted from artworks, and **Patient Input**, which may include keywords, memories, or short personal narratives. These inputs are then processed by the **Matching Engine**, which retrieves relevant images, and the **Story Engine**, which generates a personalized text based on the patient's input and the extracted dataset features. The result is a set of **Tailored Patient Stimuli**: a combination of visual and textual outputs designed to trigger memory recall and help with emotional engagement. The technical structure and implementation of each component will be explored in detail in Chapter 3.

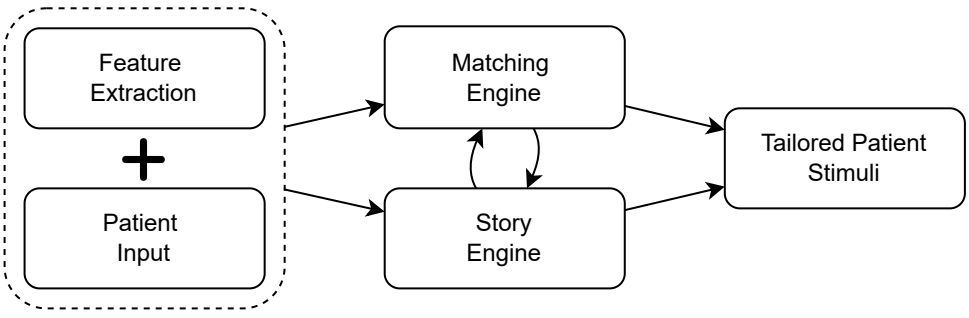

Figure 1.1: Overview of the ArtEvoke platform.

1.3 Thesis Organization

Finally, this report is organized as follows:

- **Chapter 1** (this chapter) introduces the context, objectives, and approach of the work. It provides a high-level overview of the ArtEvoke platform before diving into technical details.

- **Chapter 2** discusses the current state of the art concerning AD, available treatments, and the technological background of the AI tools used in this project.
- **Chapter 3** presents an overview of the proposed solution's design, focusing on the database architecture and the interaction modalities with the prototype.
- **Chapter 4** details the development and implementation of the prototype, delving into technical decisions and their justifications.
- **Chapter 5** analyzes the results obtained, evaluating the prototype using various metrics to assess its performance.
- **Chapter 6** concludes the thesis with a summary of the contributions, their significance, and future research directions.

To support the writing process, AI tools were used to assist in improving grammar coherence and organization in this report.

Chapter 2

Background and State of the Art

2.1 Alzheimer and dementia diseases

2.1.1 Overview

Dementia is the set of symptoms that are related to the decrease of memory, language and other logical capabilities. The tendency of this type of illness is to get worse over time, mainly with some risk factors, like age, hypertension, diabetes and other factors. The patients could include changes in mood and behavior, like personality changes and dislocation of the real world [3].

With the growth in the elderly population, there is a tendency of certain types of diseases to increase, like the ones related to dementia symptoms, which includes problems with memory, language and other logical abilities [4]. In 2021, the World Health Organization estimated the number of people living with dementia was to be around 55 million and it is expected to reach almost 150 million by 2050 [5].

AD is one that causes dementia, it is caused by damaged nerve cells in the brain mainly in the areas responsible for thinking, language and memory. It is estimated that around 60-80% of dementia in the world is caused by AD [6]. As dementia, AD is progressive, affecting increasingly the person's ability to effectively do their work and day to day activities. After some time, the disease affects motor functions, like walking, grabbing and swallowing [3, 4]. The progression of the disease is illustrated in Figure 2.1.

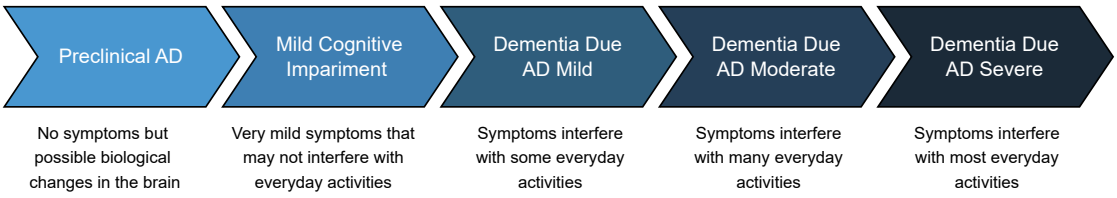

Figure 2.1: Diagram illustrating the progression of AD, adapted from [4].

The cause of AD has been studied for decades and remains a major focus of research. One of the most recognized pathological remarks of AD is the progressive accumulation of amyloid-β (Aβ) peptides in the brain. These peptides aggregate to form senile plaques (SP), which are thought to contribute to synaptic dysfunction and neuronal loss. Although the exact mechanisms remain unclear of this accumulation, approximately 95% of AD cases are sporadic, and many patients accumulate Aβ in the absence of known genetic mutations [7].

A variety of factors influence the disease, including genetics and external influences such as behaviors and habits. A 2019 study evaluated several external risk factors (e.g., high blood pressure, obesity, diabetes and depression) and their relationship with subjective cognitive decline (SCD), which refers to self-reported mental confusion. The study revealed that individuals with SCD tended to have a higher number of risk factors [8].

Patients with AD may have varying survival durations, depending on the age at diagnosis, the start of treatment, and the presence of comorbidities. The median survival time after diagnosis was 6.37 years for men and 8.81 years for women. However, when dividing by age and comorbidities, survival time ranged from 13.72 to 5.29 years for patients without comorbidities and from 6.43 to 2.98 years for those with three or more comorbidities [9, 10].

There is no current effective treatment for the progression of Alzheimer's dementia, however, some drugs have been developed to help with it. Until 2022, only six drugs had been approved by the US Food and Drug Administration. Among these, only one targets the mechanism of clearing Aβ peptides, which are considered one of the causes of the disease, while the others are aimed at alleviating its symptoms [7]. Additionally, as already mentioned, many risk factors are correlated with AD, and controlling them can reduce or delay the development of it [6, 8].

2.1.2 Cognitive assessments

Cognitive screening tools play a central role in the diagnosis and monitoring of AD and other forms of dementia. Two widely used assessments are the Mini-Mental State Examination (MMSE) [11] and the Montreal Cognitive Assessment (MoCA) [12].

The MMSE is a 30-point questionnaire that evaluates cognitive domains through both verbal and written tasks. It includes tests of orientation, attention, memory, language, and visuoconstructional skills. The first part focuses on spoken responses (maximum score: 21), while the second involves writing and drawing tasks (maximum score: 9). The test is brief, untimed, and often administered in a clinical setting where rapport is established to avoid stress or confusion [11].

The MoCA was developed to address the limited sensitivity of MMSE in detecting mild cognitive impairment (MCI). Like the MMSE, it is scored on a 30-point scale but includes more complex tasks that evaluate executive functions, abstraction, and delayed recall. It is thus considered more suitable for early-stage AD, offering higher sensitivity in detecting subtle cognitive decline [12].

2.1.3 Non-drug interventions and prevention

Alternative non-pharmacological interventions has been studied for years for dementia diseases. Most notably, diets, physical activities, and cognitive training has shown in some studies promising advantages in adopting them [7, 13].

The Mediterranean diet offers a rich eating routine, with fruits, vegetables, nuts, olive oil, and lean proteins like fish. It is well-known for its neuroprotective properties, helping to preserve cognitive functions [7]. Additionally, it addresses major risk factors for developing AD. For instance, it reduces insulin resistance, which is a primary cause of diabetes—a significant risk factor for AD [14].

As with diets, physical activities attack main risk factors of AD, such as obesity and sedentarism. The literature has shown that physical activity helps reduce the chance

of developing the disease and delays its progression in the early stages [7]. In addition, physical activity, mainly aerobic exercises, such as sports and daily tasks, improves patients' cognitive functions, leading to enhancements in their daily lives [15].

Cognitive training is a type of treatment that focuses on cognitive functions that are usually degraded by the progression of age or disease, it includes memory, language, and attention functions [13]. After completing the cognitive training, patients had small to moderate effects on cognition, overall and specific such as verbal improvements, which lasted up to one year [16]. Adaptive chunking training has led patients with mild AD to significant improvements in verbal memory performance [17].

On the other hand, Cognitive Stimulation Therapy (CST) is a more generalized approach that engages people in activities designed to stimulate thinking, concentration, and memory [13]. When compared to no cognitive stimulation in people with dementia (DA), all measured parameters showed improvement (in group or not), including cognitive function, quality of life, communication and social interaction, mood (as reported by both the patient and the interviewer), activities of daily living, and behaviors that challenge [2].

Reminiscence therapy is a form of CST that uses all the senses to stimulate memory triggers, helping people suffering from Alzheimer's disease (AD) recall their past to better cope with their present [13]. This sensory stimulation is typically achieved through familiar stimuli such as old photographs, music, scents, recorded voices, or objects from the person's past, which are used to evoke positive memories in either individual or group sessions. A three-month trial using reminiscence therapy with patients with mild to moderate AD showed positive results, including an increase in cognitive levels and a decrease in depressive symptoms [18].

2.1.4 Imagery use for intervention in dementia

With the growing interest in non-pharmacological interventions, visual arts have become a subject of study and research. While language is processed in the left hemisphere of the brain, art engages widespread neural networks due to their complexity. As a result, visual art stimulates broader brain regions, which are remarkably resilient to the brain due to aging or diseases [19].

In one study, a Cognitive Activation Therapy (CAT) was developed using art imagery, primarily aimed at addressing age-related decline and worsening cognitive function. CAT is a protocol designed to stimulate patients and support improvements in daily activities [20]. The study revealed that the length of leukocyte telomeres, a biological marker associated with cellular aging, increased significantly after the full course of CAT, suggesting a beneficial effect at the cellular level. In addition, cognitive domains such as memory, spatial perception, and communication skills showed improvement after the intervention [21].

Similarly, a culture-specific, picture-based Cognitive Training Intervention for Alzheimer's Disease demonstrated significant cognitive improvements in patients with early-stage AD. The training focused on structured visual tasks targeting key cognitive domains such as episodic and semantic memory, attention, and language, using non-verbal materials to accommodate low literacy levels. Participants showed enhanced performance in memory, attention, and language-related tasks [22]. These findings reinforce the potential of visual arts as a therapeutic tool, not only for preserving cognitive function but also for promoting neural resilience and adaptation in neuro-degenerative conditions.

2.1.5 Impact of the caregiver

Caregivers play a crucial role in the lives of people living with dementia (PLwD), often acting as a source of emotional and physical support. As studies have shown, the presence of a caregiver is associated with improved quality of life, increased safety for PLwD, and better adherence to treatment.

However, the burden that caregivers often carry comes at a significant personal cost, especially with dementia patients. One study shows that caregivers of PLwD report higher levels of stress, depressive symptoms, and health issues compared to other caregivers, highlighting the dual impact of dementia not only on patients but also on those who care for them [23]. In addition, it is mentioned that caregiving is often provided by close family members, who are usually not professionals in the field.

One of the most challenging aspects for caregivers of PLwD is dealing with neuropsychiatric symptoms (NPS), such as agitation, aggression, and emotional instability. These symptoms are often considered more disturbing than memory or functional impairments, significantly increasing caregiver stress levels [24]. In fact, NPS are frequently cited as the primary source of caregiver distress, given their emotional toll they take on those providing care.

2.1.6 AI in AD

In recent years, AI has increasingly been applied in healthcare, offering new tools for the diagnosis, prediction, and treatment of AD. Traditionally, diagnosis relied on clinical evaluations, cognitive assessments, as mentioned in 2.1.2, and imaging techniques such as MRI or PET scans. However, these methods have limitations, including insufficient sensitivity in early stages of the disease [25, 26].

AI has emerged as a promising alternative to support AD research. With Machine Learning (ML), scientists are able to detect early indicators of AD through features extracted from MRI and clinical data [25]. In contrast to ML, Deep Learning (DL) models can enhance this process by handling complex data, combining different sources of information, and providing interpretable visualizations of disease probability [26].

Moreover, DL techniques have demonstrated the ability to predict AD up to six years before clinical diagnosis based on PET scans [26]. AI has also been applied in unconventional diagnostic methods, such as analyzing retinal photographs, speech patterns, and sleep behavior, all showing high accuracy in distinguishing AD from healthy individuals or other cognitive impairments [26].

Furthermore, AI is being used to personalize treatments and enhance patient care. AI-powered cognitive stimulation tools, such as personalized brain training and voice assistants, have shown potential in slowing disease progression and supporting daily activities. These systems can adapt to the patient's cognitive level, promoting engagement and communication [26].

In addition, chatbots and virtual assistants can contribute to an improved quality of life for patients and their families [26, 27]. Recently, the use of LLMs has been explored in dementia care with promising results. LLMs can support meaningful conversation, offer cognitive stimulation, and help users with day to day activities. When integrated into easy-to-use apps, these models can adapt to individual needs, provide companionship, and reduce caregiver burden. Users and caregivers have expressed optimism about these tools, especially when designed with transparency and accessibility [27].

2.2 Large Multimodals Models (LMMs)

With the public release of ChatGPT at the end of 2022, a platform with access to the chat bot powered by the OpenAI Generative Pre-training Transformer (GPT), LLMs became widely available for users to help with their tasks. Since then, these models have been used in both professional and personal life and started a run worldwide for achieving the best results with the lowest cost. Those models have the ability of processing and generating text with human-like communication, generating an text output according to the user's input [28]. Building on this foundation, multimodal models extend LLM capabilities by processing and generating content across multiple modalities, such as text, images, audio, and video. They incorporate additional components to manage and integrate diverse modalities efficiently [29].

2.2.1 From Artificial Neurons to Transformers

One of the most influential papers in the development of Artificial Intelligence was published in 1958, when Frank Rosenblatt introduced the **Perceptron** model in his seminal paper "*The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain*" [30]. Inspired by the structure of biological neurons, the Perceptron was designed to mimic how interconnected neurons in the brain might produce intelligent behavior. In essence, it is a binary classifier that computes a weighted sum of input features and applies a threshold to decide the output: if the sum exceeds the threshold, the output is 1, otherwise, it is 0. Each input is associated with a weight and threshold which affects the decision [30]. Although limited to solving linearly separable problems, the Perceptron laid the groundwork for more complex models and is considered the first formal idea of an artificial neuron.

Utilizing the idea of the Perceptron, more sophisticated neural network architectures emerged by combining multiple artificial neurons into **structured layer**s. These networks introduced activation functions, which define how the weights and bias are interpreted. Functions such as the sigmoid, tanh, and later ReLU enabled models to learn complex, non-linear mappings [31, 32]. A basic feedforward neural network consists of an input layer, one or more hidden layers, and an output layer, where data flows unidirectionally, the simplest one with multiple layers is known as the multi-layer perceptron (**MLP**) [32]. The depth of a network, defined by the number of hidden layers, allows the network to learn representations, where successive layers increasingly capture features. This concept forms the foundation of DL.

Recurrent neural networks (RNNs) extend conventional feedforward neural networks by incorporating a recurrent hidden state, allowing them to process long sequences of data where the activation at each time step depends on the previous one. However, RNNs often struggle to learn long-term dependencies due to vanishing or exploding gradients [33]. To mitigate this issue, Long Short-Term Memory (LSTM) networks were introduced, incorporating nonlinear and more complex control mechanisms into each cell to preserve the gradient and enable effective learning across long sequences [34, 35]. Gated Recurrent Units, a simpler alternative to LSTMs, were proposed to capture dependencies over varying time scales without using separate memory cells, relying instead on gating mechanisms to control information flow [33].

However, although these recurrent architectures were designed to capture long-range dependencies, they still suffered from limitations such as vanishing gradients and long path

lengths between distant tokens. To address these issues, the **self-attention** mechanism was introduced [36]. Also known as intra-attention, it relates different positions within a single sequence to compute contextualized representations, allowing each token to consider all others in the sequence without relying on a hidden state, as in LSTM networks. This mechanism has proven effective in a variety of tasks, including reading comprehension and textual summarization [36].

At its core, an attention function maps a query and a set of key/value pairs to an output, computed as a weighted sum of the values, with weights determined by a compatibility function between the query and each key. As illustrated in Figure 2.2, this mechanism is implemented in the multi-head attention blocks of the Transformer architecture, where each token can attend to all others in the sequence. Additionally, since self-attention does not capture the order of tokens at first, positional encodings are added to the input embeddings to provide a sense of sequence. These encodings introduce information about each token's position, enabling the model to distinguish between inputs like "Augusto likes Marc" and "Marc likes Augusto" [36].

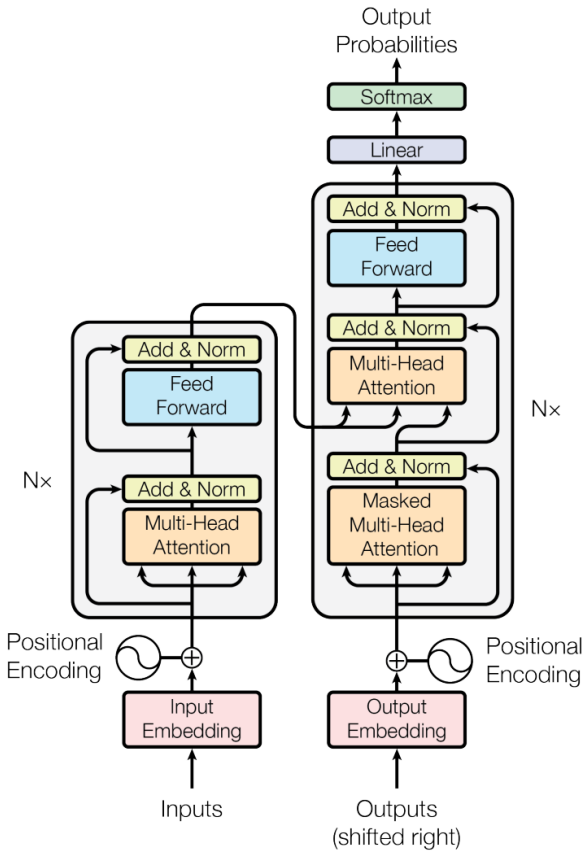

Figure 2.2: Transformer architecture. Source: [36].

Beyond being more efficient than recurrent networks, self-attention also allows for better parallel processing and makes it easier for the model to learn relationships between distant words in a sequence. Instead of passing information step by step like in RNNs, self-attention connects all words directly, shortening the "path" between them [36]. This makes the mechanism especially powerful for natural language processing tasks, where understanding the full context of a sentence is essential, even when key words are far apart.

2.2.2 Natural Language Processing (NLP)

NLP is the field that focuses on enabling machines to understand, interpret, and generate human-like language. It all began in 1966 with the development of ELIZA at MIT, a system that mimicked a psychotherapist using a rule-based approach [37]. While rule-based and symbolic methods were the foundation of early systems, the field gradually shifted toward statistical approaches in later years. Today, modern models have advanced, employing DL techniques and Transformer-based architectures to extract information from vast amounts of data [38]. Since language is inherently ambiguous and context-dependent, these models must process input in a way that captures both structure and meaning, forming the foundation for tasks such as question answering, and text summarization [38].

Before any language model can make sense of text, the input must be broken down into smaller components called **tokens**. Tokenization is the process of segmenting raw text into these units. Modern NLP systems address this with subword tokenization, splitting words into more frequent fragments. Techniques such as Byte Pair Encoding [39] and WordPiece [40] build a flexible vocabulary of subword units that balance efficiency and coverage. For example, the word “unbelievable” might be split into “un”, “believ”, and “able”. This allows the model to handle new or rare words by combining familiar parts. Tokenization is thus a key step in preparing text for DL models [41].

After tokenization, additional preprocessing steps are often applied to prepare the text for more efficient and meaningful analysis. One common step is the removal of stopwords, such as “the,” “is,” and “and” that often carry little semantic weight on their own [38]. This step is helpful in simpler NLP tasks such as document classification, where common function words do not contribute significantly.

Another common pre-processing technique involves reducing words to their root or canonical form. Stemming does that by removing prefixes and suffixes to reduce a word to its base form (e.g., "running" becomes "run"). Lemmatization, on the other hand, transforms words into their dictionary forms, considering context and part of speech (e.g., “better” becomes “good”). Both approaches aim to normalize word forms, allowing the model to treat different variations fo the same word [38, 41]. In Figure 2.3, a simple example of tokenization with stopword removal and stemming can be seen.

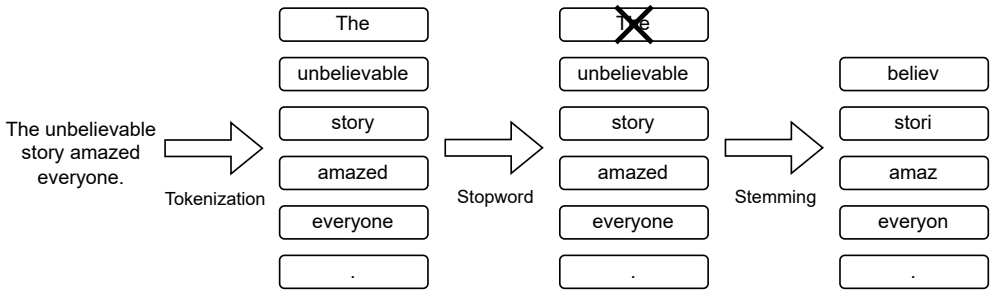

Figure 2.3: Example of techniques used in NLP.

2.2.3 LLM Architecture

The architecture is an important feature of LLMs, as it helps us understand how they process and generate text. There are three main architectures commonly used in LLMs: Encoder, Decoder, and Encoder-Decoder [29].

- **Encoder**: intended for scenarios where input processing is the most relevant part of the model, like text categorization, it uses self-attention mechanism to process the context [42].
- **Decoder**: divided into two, Causal and Prefix Decoder, both focus on the predicting of the next output token in a sequence. The first uses only previous tokens for predicting and, the second, uses bidirectional attention mechanism [42]. This architecture framework is being used in most text generation models.
- **Encoder-Decoder**: originally used in sequence-to-sequence (Seq2Seq) models based on RNNs and LSTM [34], but modern architectures, such as the Transformer [36], have become the standard for this structure. The encoder module extracts the main features and semantic attributes from the input text, while the decoder module sequentially generates output based on the context processed by the encoder.

Several commonly used models illustrate these three architectures. BERT (Bidirectional Encoder Representations from Transformers) is a pure encoder model designed for tasks like sentence classification, named entity recognition, and question answering [40]. GPT-3, on the other hand, is a causal decoder model specialized in auto regressive text generation tasks such as summarization, dialogue, and code completion [43]. Finally, BART (Bidirectional and Auto-Regressive Transformers) is an example of encoder-decoder model that perform well in Seq2Seq tasks, including translation and summarization [44].

Additionally, LLMs are typically pretrained on large-scale corpora in an unsupervised or self-supervised fashion. They are then fine-tuned on more specific datasets to adapt to particular downstream tasks. For instance, in [45], the PaLM model was fine-tuned on medical data to create Med-PaLM, which was evaluated on the MultiMedQA benchmark. As shown in Figure 2.4, the fine-tuned model significantly reduced the performance gap between the raw LLM and real physicians in medical question answering. To highlight the impact of domain-specific fine-tuning, Flan-PaLM, a general purpose, instruction-tuned variant of PaLM, was included as a baseline in the evaluation.

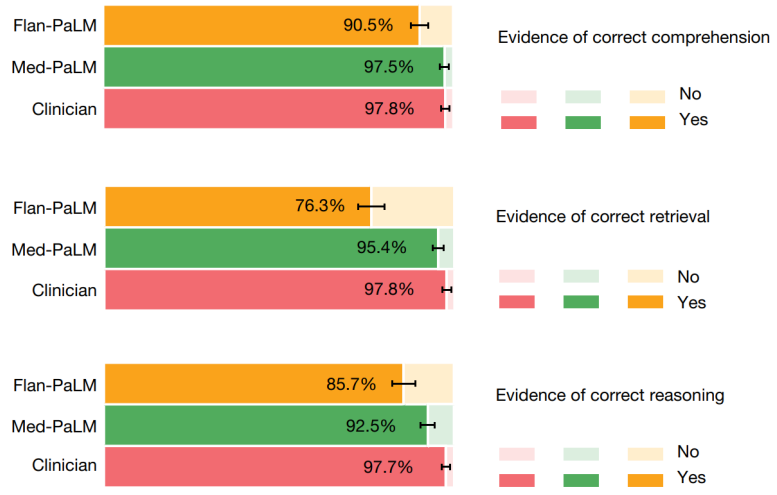

Figure 2.4: Evaluation of Flan-PaLM, Med-PaLM, and human clinicians across three metrics: comprehension, retrieval, and reasoning. Source: [45].

2.2.4 LMM Architecture

Large Multimodal Models (LMMs) require different technical approaches, as they need to process not only text but also images, audio, and video [29]. Currently, the architecture of most LMMs incorporates some, though not necessarily all, of the following five components (see Figure 2.5):

1. **Multimodal Encoder**: Extracts information from various input sources supported by the model. Examples include image encoders (e.g., ViT [46], CLIP ViT [47], OpenCLIP [48]) as well as video and audio encoders;
2. **Input Modal Aligner**: Bridges the gap between non-text features and textual tokens, facilitating the fusion of different modalities [49], like Q-Former [50], Cross-Attention Layer, and Multilayer Perceptron (MLP) [51];
3. **Pre-trained LLM**: Serves as the backbone of the model, leveraging the capabilities of pretrained language models to interpret and generate information;
4. **Output Modal Aligner**: Performs the inverse function of the input aligner, converting the unified representation from the LLM back into a format suitable for decoding into the target modality [49];
5. **Multimodal Decoder**: Generates the final output decoding features into the requested modality, such as images, or video.

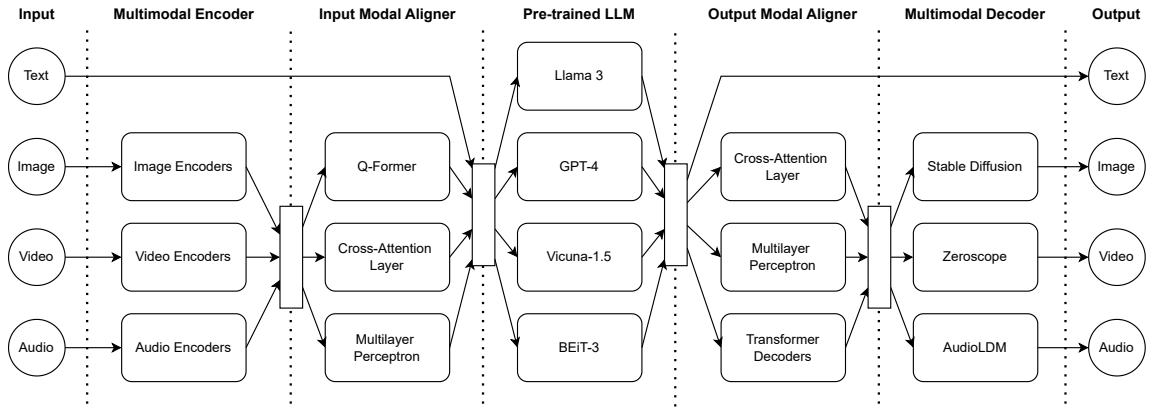

Figure 2.5: Architecture of a typical Large Multimodal Model (LMM), including components such as encoders, aligners, and decoders. Adapted from [29, 49].

Not all LMMs follow this exact architecture, it varies depending on the chosen model design or even the goal for the model. For instance, BLIP-2 [50], for example, has the capability of only generating text, so it does not have the fourth nor fifth step of the architecture mentioned. Also, Generative Adversarial Networks [52], which consist of a generator and a discriminator trained in opposition to one another, do not adopt the structure described above.

2.2.5 Benchmarks

Benchmarks are synthetic tools used to evaluate model performance under specific conditions. They allow for the comparison of models with different architectures and help

reveal the strengths and weaknesses of LLMs [53]. Currently, benchmarks for LMMs cover various aspects of reasoning, understanding, and perception, enabling users to objectively choose models that best suit their needs.

While benchmarks can provide valuable insights into model capabilities, it is important to recognize that they often diverge from real-world behavior. In particular, recent studies have highlighted the issue of Benchmark Data Contamination (BDC), where benchmark items unintentionally appear in the training data of LLMs. This contamination introduces biases that compromise the evaluation results, especially in scenarios that are intended to reflect zero-shot performance [54]. Consequently, relying on a single or solely on traditional benchmarks may yield unrealistic assessments, highlighting the need for diverse and more robust evaluation methods.

2.3 Embeddings

In this section, we will define what embeddings are, and then we will discuss how the correctness of grammar and language can affect the use of embeddings.

2.3.1 Overview

Embeddings are a concept that is mainly used in machine learning tasks relating to natural language processing (NLP). They are a way to represent items like words, sentences, documents, or even images as a set of real vectors. Thus, they project objects from a high-dimension space to a lower-dimensional one, while trying to keep the semantic information and relations intact. This allows to capture semantic relationships more easily between different items both in formats or sizes [55, 56].

Figure 2.6 illustrates how both text and images can be embedded into the same vector space. As shown, similar concepts (e.g., the word “king” and its emoji) are closer in values, while semantically distant items (e.g., a smiley emoji or a dog) appear farther apart. This visual proximity reflects their semantic similarity within the embedding space.

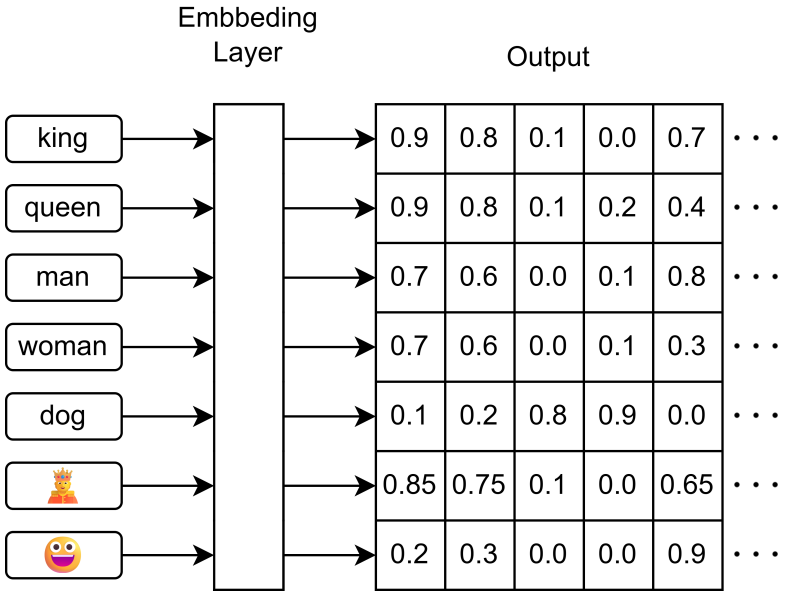

Figure 2.6: Representation of text and image embeddings in a shared vector space

In this particular example, we could interpret each dimension of the output vectors as a specific semantic feature. For instance, one could imagine that the first dimension

corresponds to “royalty”, the second to “social status”. While in real models these axes are not directly interpretable, this helps to conceptually understand how different types of items can be embedded and compared in the same space.

2.3.2 Architecture

Early embedding approaches were based on statistical or rule-based techniques, such as co-occurrence matrices or dimensionality reduction methods like Latent Semantic Analysis [55]. These methods captured basic word associations but lacked contextual understanding.

Modern approaches use neural networks to automatically learn embeddings from large text corpora. These can be broadly categorized into feature-based models, which generate reusable representations (e.g., Word2Vec [57]), and fine-tuning-based models (e.g., BERT [40], GPT [58]), which are pre-trained using self-supervised tasks and then adapted to specific downstream applications [55].

In particular, transformer-based models such as BERT produce contextual embeddings, where the same word may have different vector representations depending on the surrounding text [40]. This has proven to be significantly more effective for capturing nuance and semantic relationships than static embeddings. More recent models, like CLIP [47], go further by embedding entire sentences or text-image pairs into a shared vector space using contrastive learning objectives. These approaches are especially relevant for tasks requiring cross-modal alignment or semantic similarity at the sentence level, which are core to this project.

2.3.3 Multi-Language Embeddings

Most modern embedding models are designed to be able to understand multiple languages. However, their effectiveness can vary depending on how they were trained [59]. Models can either be trained on the same text translated to different languages, which would allow them to share an embedding space where words with similar meanings are close together no matter the language, or they can be trained by using pre-trained models for each language, and combining them together. The factors that affect the quality of multilingual models are the number of languages and the relatedness of these models [60], the training data, and the model architecture.

2.3.4 Grammatical Errors

Word embeddings are not immune when it comes to dealing with grammatical errors in the input text. In fact, it is shown that BERT’s sentence embeddings can capture syntactic relationships [61]. Knowing this, it is apparent that any sort of grammatical error would affect the output of the embeddings. Thus, any application that requires manually written text to be embedded should employ some sort of grammar check to ensure correctness.

There are many tools available that can correct the spelling and grammar of a text. In python for example, there are packages such as *language_tool_python* [62], or *pyspellchecker* [63]. These are rule-based and statistical approaches respectively. They can detect spelling mistakes in the input string, and provide suggestions for corrections.

However, tools like these are not that powerful when compared to ML approaches. For example, there are Seq2Seq models, which are usually based on RNN [64]. It can effectively correct a set of input strings. Furthermore, Neural Language Models can also

be used for this task. They work by identifying words that have a low probability of appearing in a given context, which would usually mean that they contain some sort of error. Known uses of Neural Language Models include BERT and GPT models.

2.4 Vector Database

A vector database is a specialized system designed to store and retrieve high-dimensional vectors efficiently. These vectors, typically embeddings generated by neural networks, represent data such as text, images in a way that similar items are close together in the vector space. Vector databases enable similarity search, allowing to retrieve the top most similar to a given query based on metrics like cosine similarity, Euclidean distance or inner product [65].

Unlike traditional databases, which perform exact matches on structured fields, vector databases focus on approximate nearest neighbor search (ANNS). This is essential in real-world applications where semantic similarity is more valuable than exact match. For instance, to retrieve artworks thematically similar to a query sentence, ANNS provides an efficient and relatively accurate tool, depending on the database structure [66].

One of the most widely used tools for vector search is Facebook AI Similarity Search (FAISS). It is a C++ library with Python bindings, supporting indexing methods, including brute-force, clustering-based, and graph-based techniques. FAISS is not a database per se, it does not support concurrent writes, or metadata management, but serves as the core vector indexing and search engine behind many modern vector database systems [66].

2.4.1 FAISS indexing

For smaller datasets, brute-force search is a simple and effective baseline. This method guarantees exact results and is computationally feasible on CPUs for small to medium datasets. FAISS optimizes this through efficient distance computations (matrix multiplication for batched queries) and heap structures for top-k selection [66].

However, as the number of queries or search throughput increases, non-exhaustive methods become more interesting. FAISS offers [67]:

- **IndexFlat**: Brute-force exact search and it is ideal for small datasets and CPU-only environments.
- **IndexIVF (Inverted File Index)**: This method splits the dataset into several clusters using a rough quantizer (a simple way to group vectors). At search time, instead of comparing the query with every vector in the dataset, FAISS only looks into the most relevant clusters, reducing the number of comparisons.
- **IndexHNSW**: Graph-based search structure offering strong speed-accuracy trade-offs for medium-scale datasets.
- **IndexPQ / IndexIVFPQ**: Compresses vectors via Product Quantization for memory-efficient approximate search.

2.4.2 Compression and preprocessing

Even when working with relatively small datasets (fewer than 1 million vectors), applying some compression and preprocessing techniques can still be helpful, especially

when benchmarking indexing methods or trying to save memory on CPUs. FAISS offers a few approaches with different trade-offs [66, 67]:

- **Product Quantization (PQ)** breaks each vector into smaller parts and applies k-means to each one separately. This allows FAISS to compute distances directly in the compressed form, making searches faster and less memory-intensive. PQ is often used in CPU-based setups because it offers a good balance between speed, accuracy, and memory use.
- **Optimized Product Quantization (OPQ)** improves PQ by first rotating and reordering the input vectors, making them less correlated. This step increases the quality of the compression and can lead to better recall without making the code any larger. OPQ is useful even when working with smaller datasets.
- **Principal Component Analysis (PCA)** reduces the number of dimensions in the vectors by projecting them onto the most important directions. Running PCA before PQ can make the quantizer training more stable and improve compression results, mainly if the original vectors are, for example, over 256.

2.4.3 Index considerations

FAISS provides both CPU and GPU implementations to accommodate various dataset sizes and performance requirements. The choice between CPU and GPU depends on factors such as dataset size, query batch size, latency requirements, and available hardware resources [66].

The CPU version of FAISS is versatile and supports a wide range of index types, including brute-force search (IndexFlat), inverted file systems (IndexIVF), and quantized indexes like IndexIVFPQ [66, 67]. It is particularly well-suited for small to medium-sized datasets, especially when the dataset fits in main memory, real-time performance is not a strict requirement, or GPU resources are unavailable.

FAISS's GPU implementation uses the parallel processing capabilities of GPUs to accelerate similarity search, making it effective for large-scale datasets and high-throughput applications [66–68]. It often achieves speedups of 3x to 10x compared to CPU implementations, especially when using large query batches that utilize the GPU's parallel architecture. However, transferring data between CPU and GPU memory can introduce latency, so it's more efficient to transfer the index once and issue many queries to help with this overhead.

In summary, the CPU implementation is more interesting for smaller datasets or resource-constrained environments, offering simplicity and lower overhead. The GPU implementation, on the other hand, is better suited for large-scale, performance-critical scenarios, provided that its memory and transfer constraints are taken into account.

2.5 Object Detection using AI

2.5.1 Overview

Detecting and labeling objects has long been one of the most common and challenging tasks in computer vision. However, with the development of DL models, significant progress has been made over the past decade [69]. Today, AI models can be trained to perform object detection and labeling, which means determine whether certain objects

are present in an image or video, and to identify their locations. This capability has numerous applications, including obstacle and traffic signal detection for autonomous vehicles, intrusion and facial recognition for security and surveillance, athlete and ball tracking in sports and many others.

Object detection is typically divided into two main components: image classification and semantic or instance segmentation [69]. The first refers to identifying the presence of specific objects in an image and assigning labels to them. The second focuses on locating and outlining each object within the image. Segmentation can be further divided into two key methods [70], as se in Figure 2.7:

- Semantic segmentation: assigns each pixel in an image to a predefined category;
- Instance segmentation: detects individual instances of objects within the same class, enabling not only classification of object types but also differentiation between distinct instances of the same class.

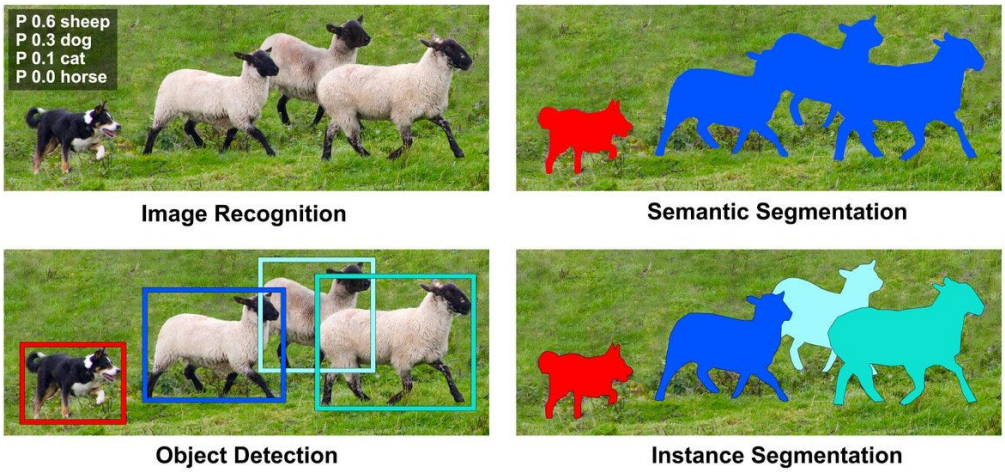

Figure 2.7: Comparison between semantic and instance segmentation. Source: [71]

2.5.2 You Only Look Once (YOLO)

One of the most popular object detection models available is YOLO (You Only Look Once). These models perform real-time object detection by processing the image only once, in contrast to other models such as Fast R-CNN which follows a two-stage process, first detecting objects, and then classifying them [72]. The YOLO detection system is simple: (1) it resizes the input image to 448 × 448 pixels, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections based on the model's confidence. This end-to-end pipeline, which can be seen in the Figure 2.8, allows for extremely fast and accurate predictions [73].

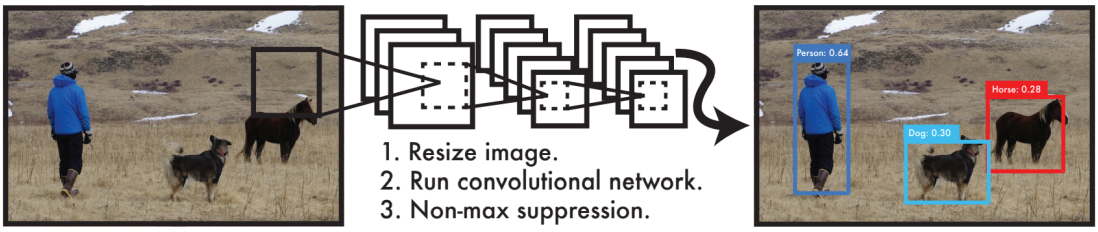

Figure 2.8: Steps used in the YOLO Detection System. Source: [73]

YOLO performs detection by passing the image through a neural network that detects multiple objects simultaneously. It distinguishes itself from other detection models by achieving both high speed and accuracy. YOLO is suitable for real-time applications or use-cases requiring high precision, as it analyzes the entire image at once, providing contextual information for better detection [73]. Its architecture is based on a Convolutional Neural Network (CNN) that divides the image into a grid. Each cell in the grid is responsible for predicting bounding boxes and class probabilities for the objects it contains. The architecture of the model is shown in Figure 2.9.

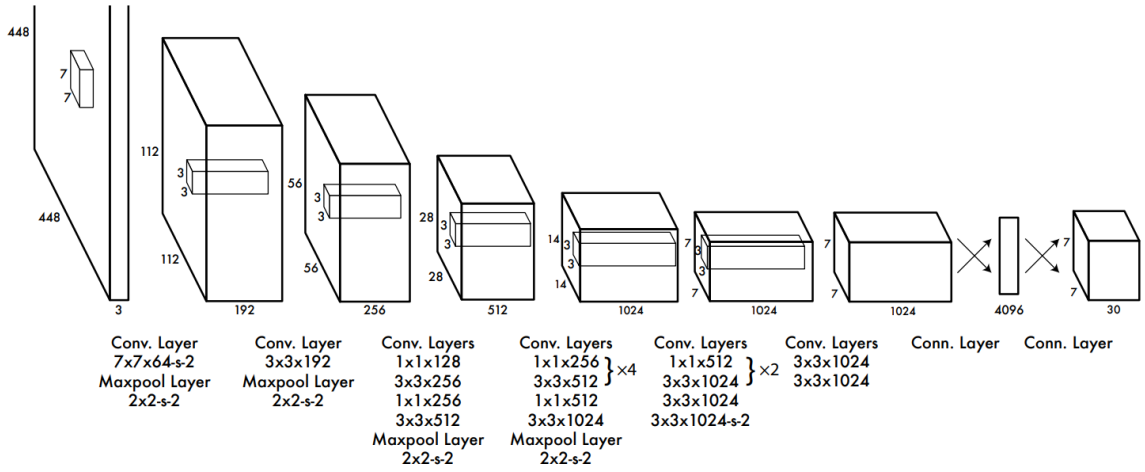

Figure 2.9: YOLO Model Architecture. Source: [73]

Since its inception, YOLO has undergone several improvements. There are now multiple versions of the model available, from YOLOv1 [73] to the most recent version at the time of writing, YOLOv11 [74]. Each iteration introduces enhancements in both architecture and performance. YOLOv11, for example, improves detection accuracy and speed by using new modules for better feature extraction, spatial attention, and multi-scale information processing [74].

YOLO models are typically pre-trained on standard datasets such as COCO, which contains 80 object classes. However, for best performance in specific applications, it is advisable to train YOLO on a custom dataset tailored to the task. This is where the concept of **transfer learning** becomes useful. Transfer learning involves taking a model that has already been trained on a large and general dataset and adapting it to a more specific task. Instead of training the model from scratch, it is possible to **fine-tune** its pre-trained weights using our custom dataset. This approach allows the model to retain useful low-level features (such as edge or texture detection) learned from the original dataset, while adjusting higher-level layers to better capture the characteristics of the new data. As a result, the model can achieve better performance with fewer labeled examples and reduced training time [73].

Chapter 3

Methodology

To create the platform's prototype, the methodology is divided into two main components: **feature extraction** and **user interaction**.

The first component focuses on extracting features from the dataset's artwork. This includes object detection, people analysis, and emotion detection. These features will allow the platform to gather meaningful information about each piece, improving the analysis for further use.

The second component addresses how users will interact with the model. Basically, how the system will actually be used. This includes two different modes of interaction with the software, which themselves will be further subdivided.

A final section details the criteria and benchmarks used to guide the selection and evaluation of the multimodal and embedding models integrated into the platform, along with the FAISS indexing architecture and the segmentation strategy used to match phrases to images.

3.1 Art Features Extraction

In this component, we aim to extract features from the art pieces in our dataset to gain a better understanding of the context or scene depicted in the paintings. Figure 3.1 shows the initial architecture designed to extract features from the art pieces.

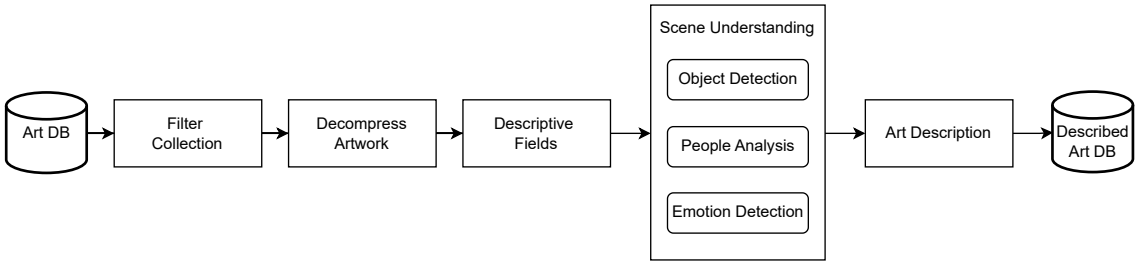

Figure 3.1: Art Features Extraction Process

The process follows these main steps:

1. **Filter collection:** We start with a dataset of art pieces from different artistic movements. For this application, we focus only on styles with realistic elements, since the ultimate goal is to match these artworks with real-life stories. The datasets used will be discussed in the Section 4.1.

2. **Decompress Artwork:** To reduce processing time and lower computational cost, the selected images are decompressed to a default, lower resolution.

3. **Descriptive Fields:** In some of the datasets used in the project, fields containing descriptive information are available. When present, such as the *iconography* field in the museum database, these fields are incorporated into the scene understanding process, as they can enhance the quality of the generated descriptions.

4. **Scene Understanding:** The images are analyzed using a scene understanding module that includes:

   - Object detection,
   - People analysis,
   - Emotion detection.

   One or more Large Multimodal Models (LMMs), as discussed in Section 3.3.2, will be tested to identify which model provides the most informative and coherent descriptions while maintaining reasonable performance. The object detection process, in particular, will be addressed in more detail in Section 4.7.

5. **Description Storage:** Once the description meets the desired quality threshold, it is stored in the database.

An additional strategy initially considered was increasing the image resolution when the description quality was below the threshold, utilizing the *iconography* and *description* fields differently from the datasets. However, after testing this approach with different LMMs, we observed that raising the resolution did not lead to a significant improvement in the output descriptions, as it will be discussed in the Section 5.1.4. As such, this method was discarded in favor of more efficient alternatives.

It is important to note that this step only needs to be run once per artwork. Therefore, even though running LMMs might be computationally expensive, since the museum's archives do not grow rapidly, the overall computational cost is not very high.

For the development of the **retrieval system**, we adopted the workflow illustrated in Figure 3.2.

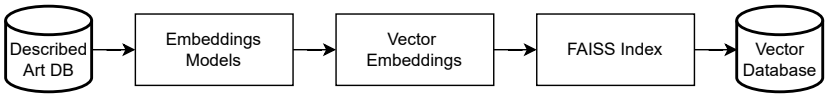

Figure 3.2: Vectorization and Indexing Workflow

As shown in the diagram, once the art descriptions are generated and stored, they are passed through embedding models to create dense vector representations. These vectors are then indexed using FAISS and stored in a dedicated vector database.

To select the most appropriate embedding model, the different models presented in Section 3.3.3 will be evaluated. Each model will be assessed in terms of computational efficiency, storage footprint, and retrieval effectiveness to determine the most suitable option for the platform. In parallel, we will also test different FAISS indexing configurations to optimize search speed and memory usage. These tests will help us balance recall performance and latency, ensuring that the retrieval system remains scalable and responsive as the database grows.

With these descriptions, vectors, and the database in place, the platform can move from feature extraction to active use. Descriptions of each art piece can now serve as inputs for the interactive modes, allowing the system to retrieve relevant artworks based on user-provided texts or prompts, as will be detailed in the next section.

3.2 User Interaction

As mentioned earlier, the interaction between the user and the platform can be done in two separate modes: **Memory Reconstruction** and **Art Exploration**.

3.2.1 Memory Reconstruction

The first mode of interaction is called **Memory Reconstruction** mode. The idea is to select a sequence of art pieces that closely correlates with the story provided by the user. Normally, the user, the caregiver or the patient themselves, is asked to write a story about a past experience or a familiar event. With the help of our platform, a sequence of artworks will be selected to help trigger the memory and stimulate different parts of the brain [19].

As shown in Figure 3.3, the workflow follows a structured process:

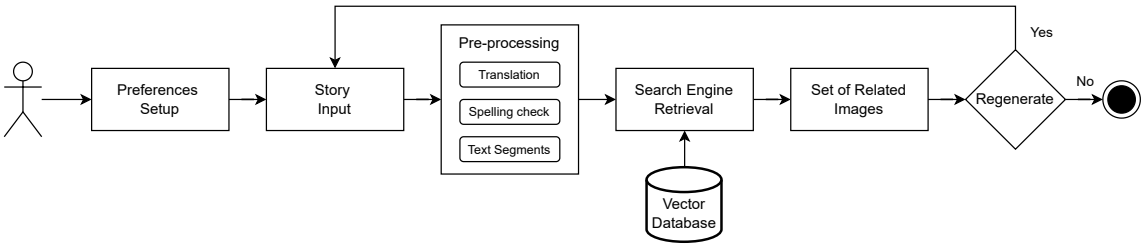

Figure 3.3: Memory reconstruction mode steps

The details of each step are described below:

1. **Preferences Setup**: Before entering the story, the user selects the preferred language and the specific dataset to work with. This ensures a personalized and context-relevant experience.

2. **Story Input**: The user writes the story about a past experience or familiar event. This story serves as the foundation for retrieving meaningful artwork that may help stimulate memory.

3. **Pre-processing**: Once submitted, the text undergoes automatic pre-processing. This includes correcting grammar mistakes, which could negatively impact model performance [61], and translating non-English input into English. The story is then segmented into semantically significant parts, studied in the next Chapter, to allow more targeted retrieval of visual content.

4. **Search Engine Retrieval**: Each text segment is converted into an embedding and compared with the stored descriptions in the vector database. This search engine retrieves the most relevant artwork for each segment based on similarity.

5. **Result Evaluation:** The resulting sequence of images is displayed to the user. These images are selected to best match the textual input and are intended to spark memory recall. For each text segment, the top-1 retrieved image is used initially.

6. **Optional Regeneration:** If the user recalls additional details or is not fully satisfied with the results, they can either provide more input or regenerate the output. If the story remains unchanged, the platform will randomly select images from the top 5 results previously retrieved. However, if any changes are made to the story, a new search is performed. This iterative process allows users to gradually reconstruct richer and more complete memories.

This iterative interaction is grounded in the idea that users may only retain fragments of a past memory or experience. By presenting them with relevant artworks throughout the process, the platform helps reconstruct these fragmented memories and encourages deeper recall.

3.2.2 Art Exploration

In this interaction mode, called **Art Exploration**, we allow an AI model to generate a story based on artworks selected by the user. The goal is to help the patient form connections between different art pieces, which may evoke forgotten memories. Figure 3.4 presents the workflow that illustrates this mode.

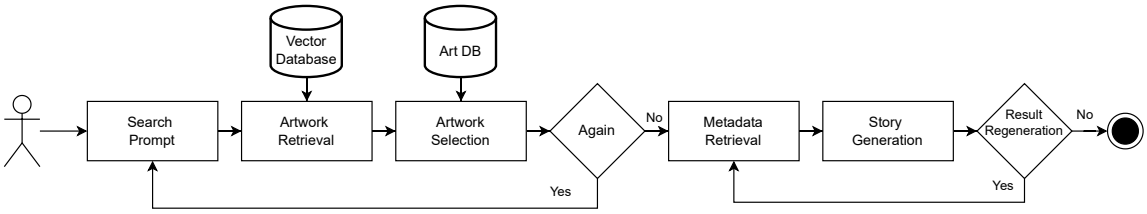

Figure 3.4: Art exploration mode steps

The workflow is structured as follows:

1. **Search Prompt:** The user provides a brief input to the system, this can be a word, phrase, or short description. This differs from the longer narrative used in the Memory Reconstruction mode.

2. **Artwork Retrieval:** A search engine queries the vector database based on the user's input and returns a selection of paintings from the art database that best match the prompt. A simplified, faster search engine may be used here to enhance interactivity.

3. **Artwork Selection:** The user selects one or more artworks from the retrieved set. If they are not satisfied, they may refine the search by submitting a new prompt, and adding more images to their artworks selection.

4. **Metadata Retrieval:** Once images are selected, additional information such as their descriptions, metadata, or the images themselves can be retrieved to provide context for the next step.

5. **Story Generation:** An LLM generates a coherent story based on the selected artworks and any integrated metadata.

6. **Result Regeneration:** The user reads the generated story. If the result is not satisfactory, they may choose to regenerate the story, potentially with a higher temperature to increase creativity, or end the process.

This mode encourages creative exploration and helps build connections between artworks and personal meaning, potentially stimulating memory through narrative construction.

3.3 Models Selection

3.3.1 Benchmarks

In this project, as mentioned, our model is expected to retrieve descriptions of art pieces, providing outputs that combine both objective and subjective elements in text format. After investigating widely used benchmarks in academia and the market, the following were chosen, in order of importance. Their use, despite the limitations previously discussed, provides a useful basis for initial comparison. For evaluating LMMs, the following benchmarks were used:

1. **Massive Multi-discipline Multimodal Understanding (MMMU)**: A multimodal benchmark designed for college-level understanding and reasoning. It features quizzes and exams from various domains and includes various image formats, ranging from paintings to diagrams [75]. This benchmark is commonly used for evaluation in recent and older models.

2. **Visual Question Answering (VQAv2)**: this benchmark assesses a model's ability to interpret visual information and answer to natural language questions, showing its understanding of the connection between visual and textual modalities [76]. Although it has more similarity with our project, the availability of this benchmark is not broad.

3. **MathVista**: This benchmark focuses on seven types of mathematical reasoning, such as algebraic and arithmetic reasoning. It includes tasks like figure question answering and geometry problem solving, all grounded in visual contexts [77].

For evaluating the quality of the embedding models used in the retrieval phase and for the vector database, the benchmarks considered were:

1. **Massive Text Embedding Benchmark (MTEB)**: This benchmark designed to evaluate the quality of text embeddings across 8 task categories, including classification, clustering, retrieval, ranking, and semantic textual similarity [78]. The average MTEB score provides a holistic view of embedding utility across diverse downstream tasks.

2. **ImageNet-1K (Zero-shot)**: A benchmark used to assess a model's ability to classify images into 1000 predefined categories without fine-tuning. The ImageNet zero-shot accuracy measures how well the model aligns visual and textual representations learned during training [79]. It provides a signal of the model's potential for visual understanding quality and text-image alignment.

For selecting the LLM used in the **Art Exploration** mode, only one benchmark was considered, as this component plays a secondary role in the overall platform and does not directly impact the retrieval or image understanding processes. The chosen benchmark was:

1. **Massive Multitask Language Understanding-Pro (MMLU-Pro)**: This benchmark is a variant of the original MMLU, covering a wide range of subjects with multitask generalization [80]. It was selected primarily because it is widely adopted and reported across the models evaluated, enabling a direct comparison of their general reasoning capabilities.

These benchmarks typically report accuracy scores ranging from 0 to 1, with 1 representing the best possible performance, usually represented by percentage.

3.3.2 Models - LMMs

After evaluating the requirements, such as access to run the model, number of parameters, and sufficient context token length, Table 3.1 presents the models selected for testing in this project for creating the description of the images. Annex A.1 includes all the models that were considered and researched, covering commercial, restricted and open-source options. All considered models are still supported, available via API or Hugging Face, and accept images as input.

Team	Name	Parameters (B)	Release Date	Context tokens (K)	Max output tokens	Weights Access
LLaVA	1.5	7	09/2023	4.096	Not specified	Open-source
	1.5	13	09/2023	4.096		
	NeXT Vicuna	7	01/2024	4.096		
	NeXT Vicuna	13	12/2023	4.096		
	NeXT Mistral	7	12/2023	4.096		
Llama	3.2 Vision	11	09/2024	128	2.048	Restricted
Qwen	2.5-VL	3	02/2025	32	8.192	Open-source
		7	02/2025			

Table 3.1: Overview of evaluated multimodal SoTA models with support for image inputs. Source: [81–87].

As mentioned, we use three benchmarks to guide the model selection process. Table 3.2 shows the performance of the shortlisted models on MMMU (zero-shot), VQAv2, and MathVista. The table with all of the model benchmarks can be found in Annex A.2.

Team	Name	Parameters (B)	MMMU (0-shot)	VQAv2	MathVista
LLaVA	1.5	7	-	78,50%	-
	1.5	13	36,40%	80%	-
	NeXT Vicuna	7	35,80%	81,80%	34,60%
	NeXT Vicuna	13	36,20%	**82,80%**	35,60%
	NeXT Mistral	7	35,30%	**82,20%**	37,70%
Llama	3.2 Vision	11	50,70%	75,20%	51,50%
Qwen	2.5-VL	3	53,10%	-	62,30%
		7	**58,60%**	-	**68,20%**

Table 3.2: Performance of multimodal models on benchmark datasets. Results are reported for MMMU (zero-shot), VQAv2, and MathVista when available. Source: [81–87].

These models will be tested and further discussed in the following chapters to determine which one is best suited for integration into the platform.

3.3.3 Models - Embeddings

After evaluating the models listed in Annex A.3, considering factors such as embedding dimension, number of parameters, date of release and other factors, Table 3.3 presents the embedding models selected for testing in this project. These embeddings will be used in two main parts: evaluating the LMM models and evaluating the embeddings for use in the application.

Team	Name	Image size	Embedding Dimension	Parameters (M)	Release Date	Modality	Weights Available
OpenCLIP	ViT-G/14 ConvNeXt-XXLarge ConvNeXt-Large@320px	256px 224px 320px	1024 1024 1024	601 848 690	Jul 2023 Jan 2023 Jan 2023	Image-Text	Open-source
Google	ViT-H-14-378 quickgelu (DFN)	378px	1024	632	Aug 2023	Image-Text	Open-source
BAAI	bge-large-en-v1.5	-	1024	335	Sep 2023	Text	Restrictive
Sentence Transformers	all-MiniLM-L6-v2	-	384	22.7	Apr 2019	Text	Restrictive
DAMO Academy	GTE-large	-	1024	335	Aug 2023	Text	Open-source
intfloat	E5-large-v2	-	1024	335	Dec 2022	Text	Open-source

Table 3.3: Overview of evaluated embeddings models. Source: [47, 88–92]

Table 3.4 shows the performance of the embeddings models selected on MTEB (Average) and ImageNet1K (zero-shot). The table with all of the models benchmark can be found in Annex A.4.

Team	Name	MTEB (Average)	ImageNet (Zero-shot)
OpenCLIP	ViT-G/14	–	80.1%
	ConvNeXt-XXLarge	–	76.9%
	ConvNeXt-Large@320px	–	79.5%
Google	ViT-H-14-378-quickgelu (DFN)	–	**84.4%**
BAAI	bge-large-en-v1.5	45.08%	N/A
Sentence Transformers	all-MiniLM-L6-v2	56.26%	N/A
DAMO Academy	GTE-large	**63.13%**	N/A
intfloat	E5-large-v2	62.25%	N/A

Table 3.4: Evaluated embedding models and their benchmark scores. Source: [88, 93]

As the LMM models, they will be tested and discussed in the following chapters.

3.3.4 Models - LLM

As needed in the **Art Exploration** mode, we integrate a LLM capable of producing narratives efficiently from the generated descriptions of each artwork. Since the descriptions are pre-generated using a multimodal model and stored in the database, this task does not require image input. Instead, it focuses solely on transforming text into narrative.

Given these requirements, the Table 3.5 shows the evaluated models selected by the authors. In Annex A.5, all the models studied for this part were included and their information.

Team	Name	Parameters (B)	Release Date	Context tokens (K)	Max output tokens	Weights Access
Qwen	3	1.7 4	04/2024	32	8.192	Open-source
Microsoft	Phi-4-mini	3.8	02/2025	200	Not specified	Open-source
Google	Gemma 3	4	03/2025	128	8192	Open-source

Team	Name	Parameters (B)	Release Date	Context tokens (K)	Max output tokens	Weights Access
Meta	Llama 3.2	3	09/2024	128	Not specified	Open-source

Table 3.5: Lightweight LLMs (<4B parameters) for story generation from text-only input. Source: [94–96]

Table 3.6 presents the benchmark results for the models selected for evaluation. A complete overview of all models considered is provided in Annex A.6.

Team	Name	Parameters (B)	MMLU-Pro
Qwen	3	**1.7**	36.76
Qwen	3	4.0	**50.58**
Microsoft	Phi-4-mini	3.8	**52.8**
Google	Gemma 3	4.0	43.6
Meta	Llama 3.2	3.0	34.3

Table 3.6: Lightweight LLMs (<4B parameters) benchmarks. Source: [87, 94–96]

These models will be evaluated to determine whether they can effectively perform the relatively simple task of composing short stories based on image descriptions.

3.3.5 FAISS

To support efficient retrieval of artworks based on text inputs, this project integrates FAISS as the core vector indexing engine. As mentioned in Section 2.4, FAISS enables fast similarity search over high-dimensional embeddings generated from artwork descriptions.

In our methodology, we compare multiple index types across both hardware setups to evaluate how these choices affect retrieval latency, memory usage, and recall accuracy. The index types were selected to cover different algorithmic strategies and compression levels.

The following FAISS index types were selected for evaluation:

- **IndexFlatIP (CPU/GPU)** – Exact brute-force search using inner product; serves as a baseline.
- **IndexIVF (CPU/GPU)** – Coarse clustering-based ANN index.
- **IndexIVFPQ (CPU/GPU)** – Inverted file index with product quantization for compressed approximate search.

3.3.6 Text Segmentation

To investigate the impact of segmentation on retrieval performance, we designed tests that evaluate how different levels of text decomposition influence the quality of retrieved artworks.

The segmentation process considers multiple levels, ranging from entire paragraphs to individual sentences. Two main segmentation strategies are explored: non-overlapping and overlapping segments. The overlapping strategy uses a sliding window approach, in which a fixed number of consecutive sentences are grouped into a segment, and the window advances (step) by one or more sentences at a time. This technique helps preserve contextual continuity across boundaries and reduces information loss.

For each segmentation level, the text is transformed into embeddings and used to query a vector database of artworks. Each of the levels mentioned is tested both with and without overlap. We evaluate the following segmentation levels.

- **Paragraph-level**: One embedding per paragraph.

- **Fixed-length groups**:
  - 3 sentences
  - 2 sentences
  - 1 sentence

The goal is to assess how segmentation granularity and context continuity affect semantic alignment and visual narrative quality. Two types of evaluation are considered:

- **Quantitative metrics**: cosine similarity between input and retrieved image descriptions, and diversity of retrieved results.
- **Qualitative assessment**: manual inspection of whether the sequence of retrieved artworks tells a coherent and relevant story.

This test informs the segmentation strategy to be used in the Memory Reconstruction mode of the platform.

Chapter 4

Development and Implementation

In this chapter, we will discuss how we developed our solution, and how we implemented it. We will include datasets, techniques and technologies that we have tried even if they are not present in the final design of our system.

All experiments and system components were developed and executed on a university-provided Linux server running Ubuntu 22.04.5 LTS. The server is equipped with an Intel Xeon Silver 4116T CPU (24 threads), 128GB of RAM, and two NVIDIA TITAN RTX GPUs, each with 24 GB of VRAM. The Python environment was managed using *venv*, and the project was implemented with Python 3.10.12. GPU-accelerated computations, such as model inference and vector indexing with FAISS, were distributed across both GPUs to reduce execution time and improve parallelism, unless stated otherwise.

4.1 Datasets

In this section, we will verify the datasets used in the project and how they were treated for each of the tests proposed for the parts of the platform.

4.1.1 SemArt

To evaluate each of the models listed in Table 3.1, a dataset containing pre-described pieces of art is required. After reviewing options, the SemArt dataset, introduced in [97], was selected. This dataset was collected from the Web Gallery of Art[1] to support the study of semantic understanding of fine-art paintings through images and artistic commentary. It includes essential information for each artwork, as it can be seen in Figure 4.1.

Figure 4.1: Information provided in the dataset. Source: [97]

[1]Available at link

The selected dataset contains over 21,000 images of artworks, with a wide range of artistic movements and themes. Given that the full dataset is excessive for the evaluation phase, a subset of 500 images was sampled, called from now on *SemArt500*. To ensure that LMMs can effectively describe a variety of images, the original distribution across types was not preserved during sampling. Specifically, 50 images were randomly selected for each of the ten different Types[2], using a *random_state* of 42. Figure 4.2 illustrates the distribution in the sampled subset, with Timeframes[3] and Schools[4] grouped for readability.

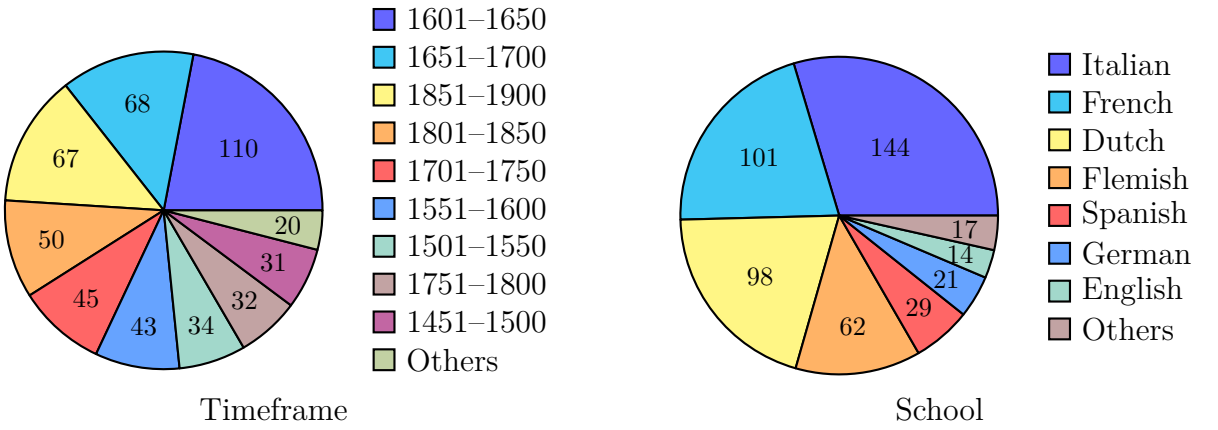

Figure 4.2: Distribution of 500 sampled artworks by timeframe (left) and by artistic school/origin (right).

For a broader evaluation of the embeddings discussed in Section 4.3, and following the selection of the final LMM model, a larger subset of images with descriptions was assembled. This expanded set includes the original 500 images, supplemented by 1,500 additional samples randomly selected from the full SemArt dataset, using the same fixed *random_state* of 42. The distribution of the *SemArt2000* is detailed in Annex B.1.

A third dataset, also sampled from SemArt, was specifically prepared to support testing of various FAISS configurations, as described in Section 4.5. To ensure the platform remains scalable, it was important to assess its performance on larger datasets. This dataset will also be used in the final version of the platform, which will be discussed in the next chapter. For this version, we excluded artworks from genres considered less relevant or detrimental to output quality due to their abstract or static nature, in the context of the platform's two main user modes. The excluded genres were *portrait*, *study*, *still-life*, *interior*, and *other*. After this filtering, the dataset consisted of **15,570** artworks, as can be seen in Annex B.2. In this setup, the original description was included in the model's prompt to help improve the quality of the generated output by providing the model with contextual information. This dataset will be referred to as *SemArt15000*.

4.1.2 WikiArt

WikiArt is a non-profit initiative aimed at making the world's art accessible. Hosting over 250,000 artworks by 3,000 artists in more than eight languages, the platform documents pieces from museums, universities, and civic institutions in over 100 countries. The project aspires to cover the entire history of art and is sustained by volunteer contributions, while advertising revenue supports technical and hosting costs [98].

Due to restricted access to the official API, we instead used a public sample of the WikiArt dataset available on Kaggle, originally used in the ArtGAN paper [99]. While the

[2]Genre, Historical, Interior, Landscape, Mythological, Portrait, Religious, Still-life, and others

[3]Others: 1401–1450 (9), 1301–1350 (7), and 1351–1400 (4).

[4]Others: Hungarian (6), Danish (5), Netherlandish (4), Russian (3), Scottish (3), American (2), Austrian (2), Belgian (2), Swiss (1), Norwegian (1), Bohemian (1), and Other (1).

original study employed the dataset for image synthesis tasks, we repurposed it to support both testing and the final deployment of our platform. This sample, referred to here as *WikiArtSample*, contains over 80,000 images organized into artistic genres.

To tailor the dataset to the platform's goals, we excluded genres that were too abstract or non-representational. Since the platform's matching mechanism relies on visual storytelling grounded in real-world experiences, abstract artworks tend to lower semantic alignment and overall retrieval quality. After this filtering step, we retained around **51,928** artworks across 14 genres. This sample will be called *WikiArtSampled*. The distribution of images per genre is illustrated in Figure 4.3, be aware that an artwork can be part of more than one genre.

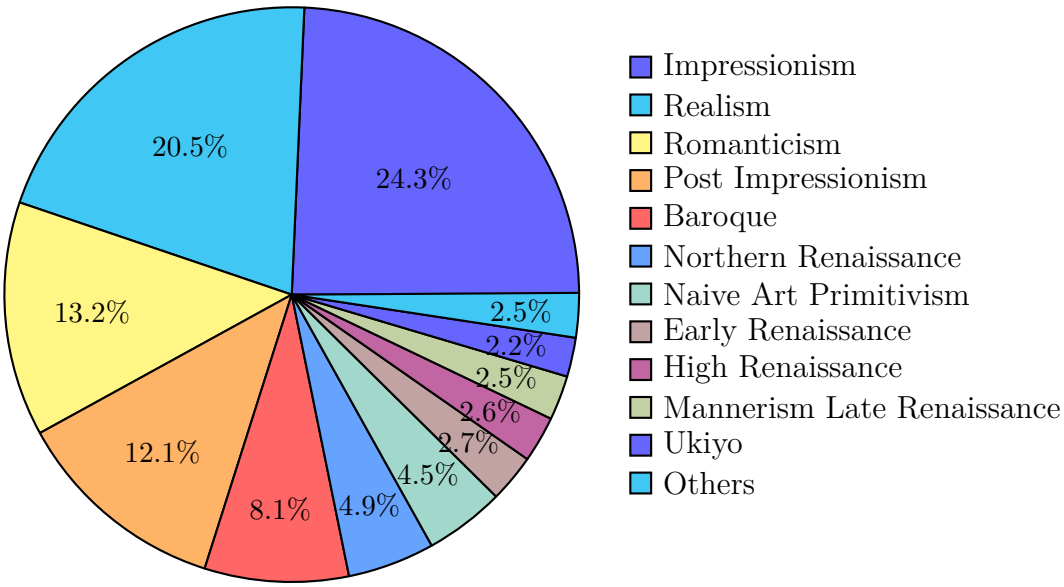

Figure 4.3: Distribution of retained artworks in the *WikiArtSample* dataset by genre.

4.1.3 Royal Museum Archive

As this project is conducted in collaboration with the Royal Museums of Fine Arts of Belgium in Brussels, we received a dataset of artworks from their collection to be integrated into the prototype. The dataset includes approximately **6,872** high-quality images and will be called *RoyalArchieve* in this report. However, the metadata provided is limited, containing primarily the artist's name and some descriptive fields of each work.

Figure 4.4 shows the distribution of artists based on how frequently they appear in the dataset. In total, there are 1,718 different artists. Most of them appear only once, and only a small number have more than 50 artworks associated with them.

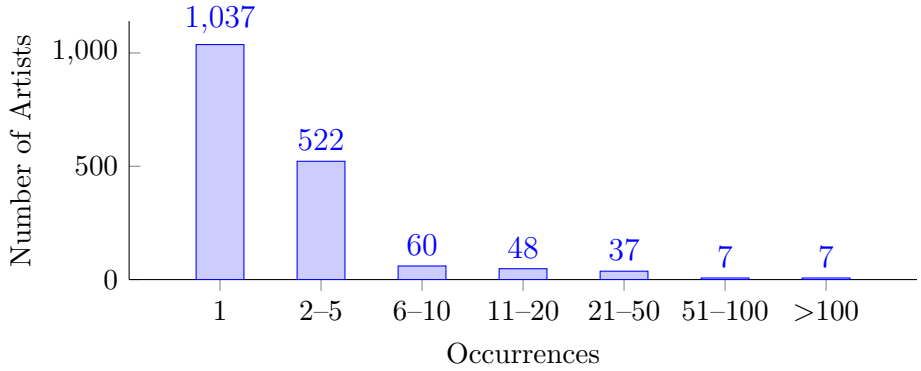

Figure 4.4: Distribution of artists by number of occurrences (bar chart).

Although only a few artists have more than 50 works in the dataset, they collectively represent a substantial portion of the collection. As illustrated in Figure 4.5, just 14 artists account for approximately 2,174 artworks. The first two alone, Constantin Meunier and Antoine Wiertz, already make up a significant share of the total, more than 1,000 images.

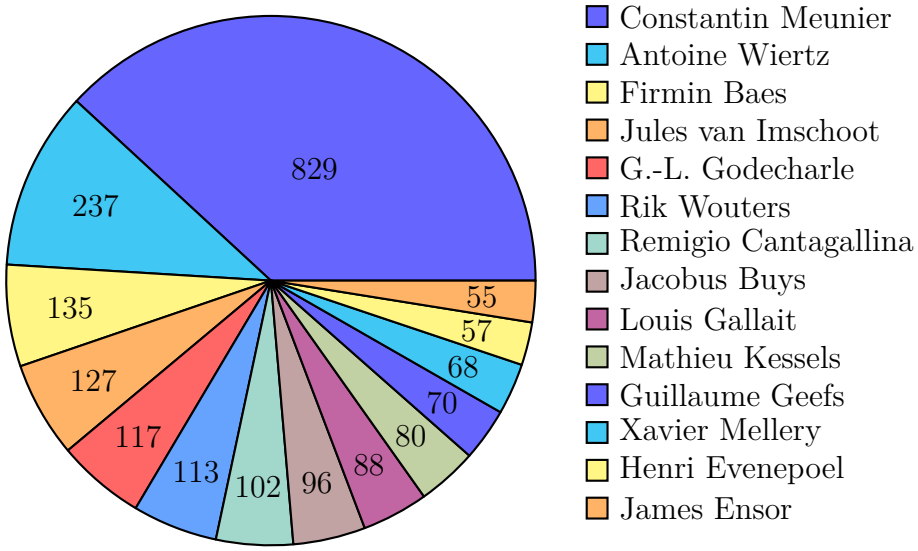

Figure 4.5: Distribution of artworks in the *WikiArtSample* dataset by artist (top 14).

In addition to artist and title information, the dataset includes detailed iconographic descriptions for some artworks. This metadata typically lists visual elements (such as objects, symbols, and characters), the type of painting, and occasionally interpretive commentary. When available, such data is particularly valuable for providing the LMM with initial cues and contextual hints, and, consequently, improving the quality of the generated descriptions.

An example of iconographic annotation is provided in Box 4.1.3, illustrating how this information is structured in the dataset.

Iconography for artwork: *"Instruments de musique"*

Subject Terms: table ; tapis ; partition ; instrument de musique : viole de gambe ; guitare ; cistre ; mandore ; viole ; luth ; flûte ; bois ; ivoire

Specific Subject Identification: Nature morte avec instruments de musique

Iconographic Terms: nature morte ; musique

Iconographic Interpretation:

Plusieurs instruments de musique sont disposés sur une table recouverte d'une nappe, qui laisse apparaître un tiroir ouvert duquel sort une partition musicale ; la basse de viole au centre est entourée de deux guitares en marqueterie de bois et ivoire. À l'avant-plan à droite, deux petites pommes en voie de décomposition. À l'arrière-plan, on peut voir un luth, une flûte. Les instruments abandonnés à la fin d'un concert et devenus muets, la présence à droite de deux petites pommes en voie de décomposition et le silence qui hante le tableau concourent à évoquer la précarité et la fugacité de la vie. Il y a ici tous les symboles d'une *Vanitas* ou d'un *Memento mori* (d'après Catherine Heesterbeek, in *Musée d'Art Ancien. Oeuvres choisies*).

Conceptual Terms: Vanitas ; Memento mori

It is important to note, however, that iconographic data is not available for all artworks, and the level of detail varies across entries. Also, the dataset provides other metadata,

such as the physical dimensions of each artwork and two image versions per piece, one in high resolution and one in low resolution. However, only the high-resolution version was used in this project.

4.2 LMMs evaluation

As mentioned in Section 3.3.2, several multimodal models were selected for evaluation. The main objective is to find an effective way to describe artworks in text format. This is crucial, as the generated descriptions will be used to match artworks with sections of the story content provided by the user. The evaluation relies on quantifying the similarity between the written description and the image itself. Therefore, selecting the model that best fits our application is of utmost importance.

4.2.1 Evaluation protocol

To standardize evaluation, the prompt shown in Box 4.2.1 was used for all models. This prompt includes an example of a "good description," which serves to guide the model's output toward the desired descriptive style found in the selected dataset. Therefore, we evaluate the models based on their one-shot description generation, using the provided example as context.

Prompt Given

Task: Describe the visual content of the following artwork in detail. Your description should focus only on what is visible in the image, such as people, objects, colors, emotions, actions, setting, and atmosphere. Do not include the title, artist, art movement, or historical context.

Example of a good description: "The lunette on the back wall of the loggia depicts a colorful fruit and vegetable market. The scene is split in two: women run their stalls beneath a tall arcade supported by a striking red pillar. Besides produce, other goods hang from a molding along the back wall."

Now, describe the current image following the same style. Be accurate, descriptive, and grounded in what can be seen.

4.2.2 Quantitative evaluation

To evaluate the quality of the descriptions generated by each model, we used embeddings and compared them using cosine similarity. This metric measures the cosine of the angle between two vectors in an inner product space, producing values between –1 and 1, where higher values indicate greater similarity. It is computed according to Equation 4.1:

$$cosine\_similarity = \frac{A \cdot B}{|A| \cdot |B|} \tag{4.1}$$

Where:

- A, B: n-dimensional vectors (e.g., model embeddings);
- $A \cdot B$: dot product between vectors A and B;

- $|A|$, $|B|$: Euclidean norms of vectors A and B.

In practice, when embeddings are normalized to unit length (i.e., $|A| = |B| = 1$), cosine similarity simplifies to a dot product between the two vectors, making the computation more efficient. In our setup, all embeddings were normalized after being generated by the embedding models.

To assess how well the generated descriptions matched the originals, we used cosine similarity between their respective embeddings. These embeddings were produced using four text-focused models: bge-large-en-v1.5, all-MiniLM-L6-v2, gte-large, and e5-large-v2. We chose these models because they perform well on tasks that measure how well a model captures the meaning of text, as shown by their scores on the MTEB benchmark (see Table 3.4). Among them, gte-large and e5-large-v2 stood out with the highest scores, at 63.13% and 62.25%. Although all-MiniLM-L6-v2 has a lower score (56.26%), it's known for being fast and lightweight, which makes it a popular choice. We also included bge-large-en-v1.5 for its growing use in multilingual and specialized retrieval tasks. Overall, this mix of models gave us a robust and diverse basis for evaluating how the generated descriptions matched the originals.

This experimental step not only aims to ensure that the chosen resolution does not constrain the model's performance but also supports the proposed Art Feature Extraction workflow. In this workflow, an increase in resolution will be tested to assess whether it improves the quality of the generated descriptions. The complete results of these resolution-based evaluations are presented and analyzed in Chapter 5.1.

4.2.3 Resolution analysis

Following the selection of the model most suitable for the application, additional experiments were conducted to assess the potential impact of input image resolution on the quality of the generated descriptions. The underlying hypothesis was that higher-resolution images, by preserving more visual detail, could enhance the model's ability to produce accurate and contextually rich descriptions. To evaluate this, various image resolutions were tested during inference. For each configuration, the generated descriptions were compared to their corresponding reference descriptions using cosine similarity, in accordance with the methodology employed in the primary evaluation.

4.3 Embeddings models evaluation

4.3.1 Evaluation protocol

To evaluate the embeddings, we need to assess how effectively and correctly the selected embedding models can transform the content into a vector representation. This enables us to later determine how well the model can match it to a story provided by the user. To simulate the behavior of the platform, we created a test scenario using a FAISS vector database built with *SemArt2000*, mentioned in Section 4.1.1. Each image is represented by both its original description and the description generated by the model chosen.

To better reflect the actual platform behavior, we focused solely on the generated to original description strategy. In this setting, the FAISS database is built from generated descriptions (as they will be in the final version), and queries come from original human-written descriptions. This setup relates with the platform's need of retrieving relevant artworks based on user input.

To initiate the evaluation, we recorded the time each embedding model required to encode the 2,000 generated descriptions and, when applicable, the 2,000 corresponding images. The total duration shown reflects the time to embed all descriptions at once, using a single GPU. For models that support image input, the images were processed in batches of 100 to manage VRAM constraints. This comparison helps us assess the computational efficiency of each model for both text and image embedding tasks.

4.3.2 Quantitative evaluation

For the main evaluation, we used the **Recall@K** metric with $K = 1, 3, 6$, which measures the percentage of times the correct image is retrieved within the top-K results. This allows us to assess how well each embedding model ranks the correct image based on an original description query. To perform this retrieval efficiently, we used the FAISS library to build and query the index for each embedding model. Specifically, we used an IndexFlatIP index, chosen for its speed, exact search and simplicity. Since all embeddings were normalized, using the inner product (dot product) in this setup is equivalent to cosine similarity. While these results are based solely on text embeddings, some models also support generating image embeddings in the same vector space. To test whether using image embeddings in the index could improve retrieval, we conducted an additional experiment combining both modalities.

4.3.3 Qualitative evaluation

While Recall@K is useful to assess whether the exact target image is retrieved, it doesn't capture the semantic relevance of the results returned. The primary goal of the platform is not to find the original image, but to retrieve artworks that are similar in theme, content, or visual style of a certain part of a text. Therefore, even if the original artwork is not present in the top-k, the result can still be considered successful if the retrieved images are contextually appropriate.

To take this into consideration, we conducted a qualitative evaluation in which subjective judgment was used to assess the top-k results retrieved by each model. For three randomly selected query images, presented in Figure 4.6, we manually reviewed the top 10 retrieved images for each model and assigned a binary relevance score: 1 if the image was considered semantically relevant to the query description, and 0 otherwise.

(a) Village scene by Pieter Brueghel. (b) Horse gallop painting by Géricault. (c) Market scene by Louise Moillon.

Figure 4.6: Side-by-side images used for evaluating subjective similarity.

The purpose of this qualitative evaluation was to narrow down the number of candidate models for final deployment. By assessing the semantic relevance of the top retrieved results, we were able to identify the most promising models, which were then selected for further

evaluation through a user survey. This step ensured that only the top-performing models, based on both quantitative and qualitative criteria, were considered in the subsequent phase of subjective analysis.

4.3.4 User survey

To further investigate the evaluation of the embeddings with a subjective view, we selected the two most promising models for a broader subjective evaluation. The main idea is that the models were chosen based on their complementary strengths, one excelling in Recall@K, the other in a solo-human-judged thematic alignment.

By conducting a user survey comparing these two models across more diverse queries, we aim to explore how subjective perceptions of relevance align with algorithmic retrieval and gain insights into the models' real-world usability from samples more representative. The survey was structured to assess both the accuracy of top-ranked results and the consistency of high-quality retrievals.

Each participant was presented with a set of six descriptive phrases, shown in Box 4.3.4. These phrases were inspired by well-known literary works and were designed to reflect themes and scenes likely to be represented in the dataset used in this study.

Descriptive Phrases

Inspired by *The Great Gatsby*
A grand mansion stands by the water, filled with lights and music, as elegantly dressed guests arrive under the moonlight.

Inspired by *A Song of Ice and Fire*
At its highest point, a great castle looms over the rooftops, as the king's procession winds through streets lined with watchful citizens.

Inspired by Shakespeare's *Hamlet*
On a hillside, peasants gather in quiet clusters, their tools resting beside them as guards ride past toward the distant walls.

Inspired by Tolstoy's *War and Peace*
A dusty path approaches a village, where chickens scatter and children pause their play to watch him pass.

Inspired by Salinger's *The Catcher in the Rye*
A wide rye field stretches toward the edge of a cliff, as children run through the grass while a figure watches over them from a distance.

Inspired by Bradbury's *Fahrenheit 451*
A firefighter stands before a wall of burning books, the orange glow lighting up his soot-covered face as ash rains down in a quiet suburban street.

For each description, the evaluation included three comparisons between the models:

- **Comparison 1**: Top 1 vs Top 1
- **Comparison 2 and 3**: Random sampling from Top 5

In each comparison, participants were shown two images side by side, each retrieved by a different model, and were asked to rate their preference on a scale from 1 to 10. A

rating of 1 indicated a strong preference for the left image, while 10 indicated a strong preference for the right image. This approach allowed us to capture nuanced differences in how each model aligns with human perception of the descriptions.

The first comparison tested the models' ability to rank their most relevant result first. The following two comparisons, based on random selections from each model's top-5 results, aimed to simulate a more dynamic, user-driven interaction, similarly to using the "regenerate" button to explore alternative matches. These tests helped assess how consistently each model returned semantically appropriate content across its top predictions.

All image pairs were anonymized and their display order was randomized to minimize positional bias.

4.4 LLM Evaluation

4.4.1 Evaluation Protocol

To evaluate the different LLMs presented in Section 3.3.4, we conducted a focused experiment using the lightweight models selected. Since the story generation component is significantly lighter than the multimodal description step, the selected models are expected to run in a period that doesn't disrupt the user experience. Three representative images were selected for testing, as shown in Figure 4.7.

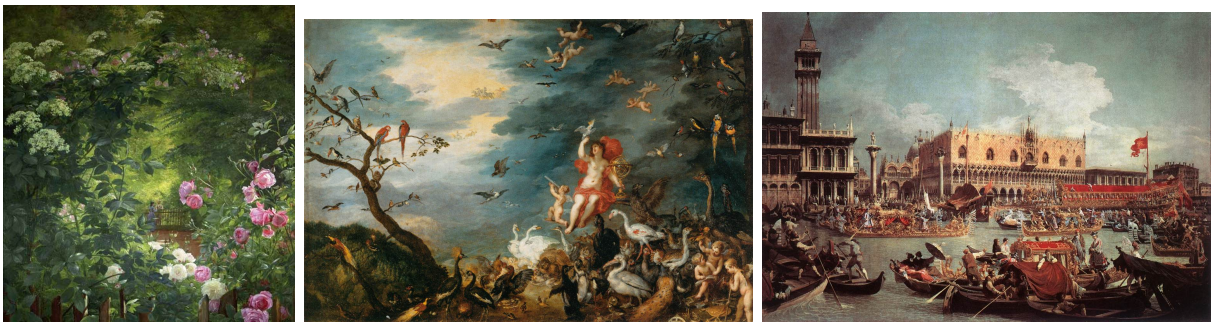

Figure 4.7: Images whose description was used for testing LLMs.

Each model was prompted with the same text input, illustrated in Box 4.4.1, which includes the generated descriptions for the image. The task of the model is to generate a short story that logically integrates and builds upon the provided descriptions.

Test Prompt

Descriptions:

Art descriptions will be inserted here

Write a story that takes inspiration from these scenes. Use 2–3 short paragraphs (approximately). Tell it like a simple, flowing story with a start, middle and an end. The paragraphs have to be connected and follow a sequence of events.

Each model is evaluated using the same prompt to account for variation in generation time and potential randomness in output. All tests were conducted under the same hardware conditions.

4.4.2 Quantitative & Qualitative Evaluation

The following metrics were used to assess model performance:

- **Average Generation Time (in seconds)**: The average time taken to generate a story, computed over 10 runs per image, both using the GPU and the CPU.
- **Story Success Rate**: The proportion of successful outputs that result in a coherent story. Outputs were considered successful if they included at least two narrative sentences logically connected to the descriptions.
- **Average Quality of Story**: A subjective evaluation performed by the authors, assessing how well the generated story flows, its coherence, and narrative quality. This score ranges from 1 to 5, if the story is successful.

These metrics help guide the selection of the most suitable LLM for integration into the **Art Exploration mode**.

4.5 FAISS Evaluation

4.5.1 Evaluation protocol

To better understand how different FAISS indexing strategies and hardware setups perform, we tested four representative index types using the *SemArt15000* dataset, a set of more than 15,000 artworks with its generated descriptions, as introduced in Section 4.1.1. All descriptions were converted into embeddings using the model chosen in the earlier step in the project.

Even though 15,000 images are relatively few for FAISS, which is optimized for datasets with millions or billions of vectors, our goal wasn't only to test scalability. Instead, we wanted to explore the trade-offs in latency, memory usage, and retrieval quality that are most relevant to our current application. This also gives us a clearer idea of how the system might behave if the platform grows or faces resource limitations in the future.

4.5.2 Quantitative evaluation

We evaluate each method based on:

- **Search latency** (average time per query),
- **Retrieval quality** (**Recall@K** metric with $K = 1, 3, 6$),
- **Memory requirements**.

The six FAISS index configurations mentioned in the Section 3.3.5 will be tested with these metrics. This selection covers a mix of exact and approximate methods, and contrasts CPU and GPU-based setups to analyze the impact of hardware on performance.

4.6 User Input Pre-Processing

4.6.1 Spell-Checking

As discussed in Section 2.3.4, grammatical errors can negatively affect the quality of embeddings. Therefore, it is essential to verify and correct user input before evaluating its semantic content. The first step in this preprocessing pipeline is spell-checking. After reviewing several available tools, we selected the Python library provided by LanguageTool [100]. This open-source library is widely used across different sectors and offers robust

multilingual support, covering more than 20 languages. Its effectiveness in detecting grammatical errors has been demonstrated in previous research [101], and it is even integrated into commercial applications such as Microsoft Office [100].

4.6.2 Multi-Language Support

The next step in the preprocessing pipeline involves handling multilingual input. Since the web application is intended for a diverse user base, it is important to support text input in various languages. However, the selected embedding model is designed to work only with English text. To address this limitation, we integrated a translation step into the pipeline.

After the initial spell-checking, the input text is passed through a translator powered by the deep-translator Python library [102]. This open-source tool offers flexibility by supporting multiple translation engines and a wide range of languages. It also has the advantage of working offline and maintaining high processing speed, making it a practical choice for real-time applications. These features, along with its ease of integration, motivated its selection for this project.

As a result, all user input is automatically translated into English before being processed by the embedding model.

4.6.3 Text Segmentation

We selected a simple narrative divided into three paragraphs, shown in Box 4.6.3. This text was segmented according to the strategies outlined in Section 3.3.6. The *SemArt15000* dataset was used for retrieval due to its larger scale, which enhances the diversity and representativeness of the retrieved results.

A Walk Through the Village

I remember arriving in a small village by train while it was raining. The station was quiet, just a couple of people waiting, and a bike leaning against the wall. I had a small bag with me and a letter in my pocket with an address. A dog was sleeping under the bench, and the only sound was the soft hum of the engine as the train pulled away.

Instead of going straight to the house, I took a walk through the village. The streets were narrow and made of stones. I passed a small shop with flowers in front, and a boy kicking a ball by himself. There was a little church with a bell that rang as I walked by. Everything felt calm, like time moved slower there.

When I got to the house, no one was home yet. I sat on the front step and opened the envelope I had brought. Inside was a postcard from my girlfriend at the time. She was smiling in the photo, standing in a wide field with hills in the background and flowers all around. On the back, she had written, “Wish you were here. Enjoy your vacations in your uncle’s house.”

Two distinct configurations were implemented for sentence-level segmentation: **non-overlapping** and **overlapping (sliding window)**. In the non-overlapping configuration, fixed-size sentence groups are created such that each sentence appears in exactly one segment. On the other, the sliding window strategy produces overlapping segments by advancing the window or step, one or two at a time. This last setup helps to preserve context and mitigate semantic fragmentation.

For each segmentation strategy, the resulting segments were embedded using the model identified as the most effective in earlier evaluations. All queries were performed using a

FAISS index with the IndexFlatIP configuration on GPU. The top-1 most similar image was retrieved for each segment based on cosine similarity, generating a total of seven distinct image sequences, discussed more in the Section 4.3.

The following evaluations were carried out to assess the performance of each strategy:

- **Cosine similarity**: average similarity between each segment and its retrieved image description.
- **Diversity score**: total number of unique images retrieved across the full sequence.
- **Qualitative review**: manual assessment of narrative coherence and semantic alignment between the story and retrieved images.

The outcomes of these evaluations are presented in Section 5.5.

4.7 Object Detection Model

Regarding the object detection and labeling model, we decided to use a YOLO [73, 103] as mentioned in the Section 2.5.2. YOLO is a popular object detection and image segmentation model, which is already pre-trained on a dataset. Developers can also apply transfer-learning on the model by training it on their own dataset to specialize it for their task. For this project, we decided to use the latest version available, YOLOv11 [74].

4.7.1 Model Training

We fine-tuned the model on the *WikiArtSample* dataset, as explained in Section 4.1.2. As in Kellenberger et al. [104], we labeled only a portion of the data. That work describes the use of active learning and suggests that labeling as little as 0.5% of the dataset can be sufficient for effective model training. Inspired by this approach, we applied a similar strategy to reduce manual labeling effort while maintaining training efficiency.

With that information in mind, we began by creating a training set containing 0.5% of the total images from our dataset. This subset was manually labeled and then split into training, validation, and testing sets. We used it to fine-tune the YOLO model, experimenting with different image sizes, numbers of epochs, and early stopping criteria to optimize performance.

After this initial training phase, we implemented an active learning loop. The fine-tuned model was used to perform object detection on the remaining unlabeled data, returning the 20 images with the lowest confidence scores. These images were then manually labeled and added to the training set. This process was repeated iteratively until the model reached a satisfactory performance level. The steps of the training process are visually summarized in Figure 4.8.

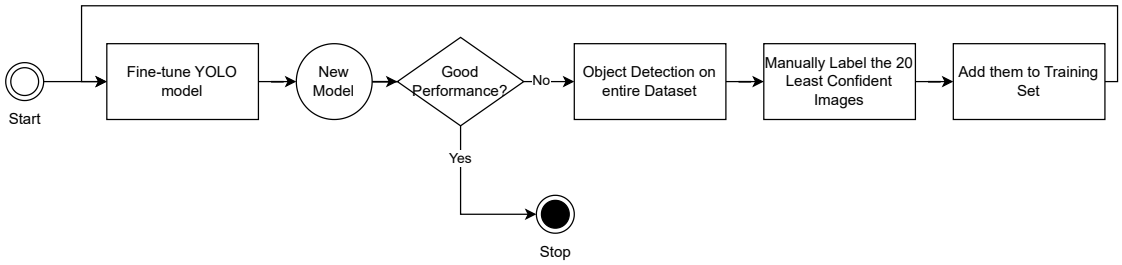

Figure 4.8: YOLO Training Process

4.7.2 Evaluation Protocol

To evaluate the object detection model's performance, we used metrics such as mean average precision (mAP), intersection over union (IoU), and bounding box regression loss (box loss). These respectively represent the average precision of predicted bounding boxes per class, the ratio of the area of overlap to the area of union between predicted and true bounding boxes, and the loss indicating how closely the predicted box matches the actual one. These metrics were calculated after each iteration of the training loop.

During evaluation, however, we encountered a major issue: a strong class imbalance in the training labels. The dataset initially contained 105 different classes, but a large number of them accounted for less than 5% of the total labeled instances. Some classes had only one example. Meanwhile, the class "person" appeared around 1,000 times, significantly dominating the dataset. The second most frequent class, 'tree', appeared only around 300 times. This imbalance is illustrated in Figure 4.9.

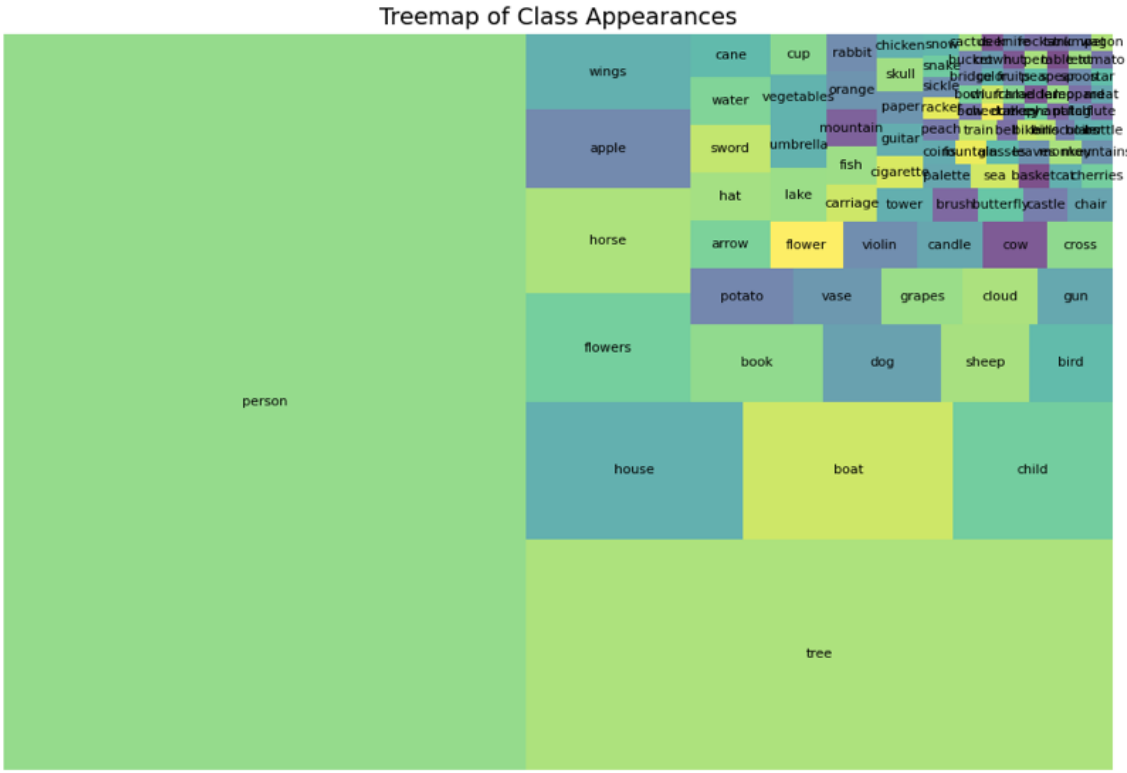

Figure 4.9: YOLO Class Frequency Map

This issue impacted the model's ability to generalize, as it quickly overfitted to the "person" class and misclassified many other objects. To mitigate this, we implemented several strategies. First, we computed class weights to identify underrepresented labels and applied six types of data augmentation to images containing these rare classes. We also removed classes that only appeared once, reducing the number of classes to 43.

Despite these efforts, class imbalance remained a significant problem. While these mitigation steps were integrated into our training process, the results did not reach an acceptable performance level for inclusion in the final platform. A detailed analysis of the performance metrics and our final decision is presented in Section 5.6.

4.8 Interface

For the user interface, we decided to create a web application, along with a database to store user credentials, which would allow us to implement additional features tailored to each user. This will be discussed later in this section. It is important to note, however, that creating an account is not required to access the core features of the application, as we prioritize accessibility for all users. An overview of the home page of the web application is shown in Figure 4.10.

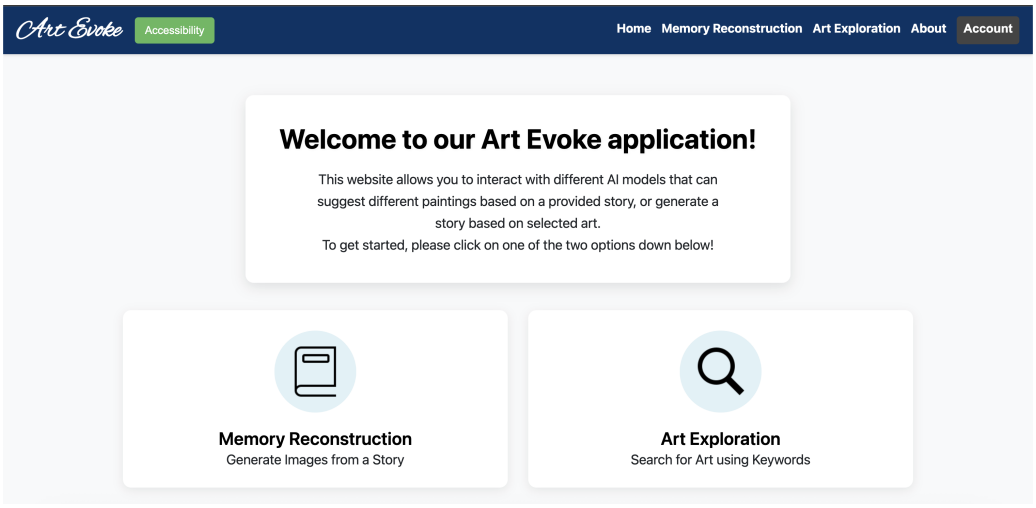

Figure 4.10: Interface Home Page

There are two main sections in the web application: the Story Generation page and the Art Search page. These correspond to the two main use cases of the project described previously.

4.8.1 Database

As mentioned in the previous section, we designed a database that can store user credentials in order to add the functionality of having accounts. Since the database is simple and does not have many entities or relations, we decided it would be easier to use a NoSQL database, and we specifically chose MongoDB [105]. With this database in place, the users can create users which would allow them to save anything they generated in the application to their account, allowing them to view the art and/or stories they have created.

4.8.2 Backend

When developing the backend, we needed to ensure that it could run efficiently on a GPU server, as the AI models we developed require significant computational resources for inference. Without such a setup, the user experience would be negatively impacted due to high response times when using model-based functionalities. To address this, we chose to use FastAPI [106], a modern Python web framework that integrates seamlessly with Python-based AI models and allows us to fully leverage GPU capabilities for faster and more responsive performance.

4.8.3 Frontend

For the development of the frontend, we decided to use ReactJS [107]. It is a great JavaScript framework that provides modular architecture, and is a great choice to build

complex and large applications. While the app is currently simple, it is crucial to have the foundation for developing a scalable application if needed in the future.

4.8.4 Accessibility

Overview

As the project is aimed at aiding people suffering from Alzheimer's, it is crucial for the web application mentioned above to meet accessibility standards. In order to do so, we will follow some conventions mentioned in the Web Content Accessibility Guidelines (WCAG) [108]. The latter was created by the World Wide Web Consortium (W3C), which is the main entity defining international standards involving the World Wide Web.

In order to make the accessibility features easy to access for any user, we created an accessibility panel that consolidates most of these features. As shown in Figure 4.11, this panel can be accessed via a button located in the navigation menu of the web application, making it available on every page.

Figure 4.11: Interface – Accessibility Panel

Perceivability

One of the most important aspects is font size, as elderly users may have reduced visual acuity. It is essential that they are able to resize the website content. To address this, a font size slider is included on the homepage, allowing users to scale the entire application up or down. This feature aligns with principle 1.4.4 of the WCAG. An example of this slider is shown in Figure 4.12.

Figure 4.12: Interface – Font Size Slider

Operability

There are many standards involving the operability of the web application. For example, principle 2.1 and all of its sub-principles require that the entire interface is accessible

with a keyboard. We have ensured that it is indeed the case as the user can navigate through the pages, elements, and access all of the functionality solely with the use of the keyboard, and there are no trap elements that would block the user from exiting them without another input method.

Furthermore, guideline 2.2 is respected, as there are no time-sensitive events within the application, users can take as much time as they need when interacting with the website. Guideline 2.3 is also met, as the application contains no flashing lights or animations that could trigger seizures or cause adverse physical reactions.

The website allows for different input modalities by offering a speech-to-text alternative where needed, as illustrated in Figure 4.13. This feature does not restrict the use of other input forms and adheres to guideline 2.5 of the WCAG.

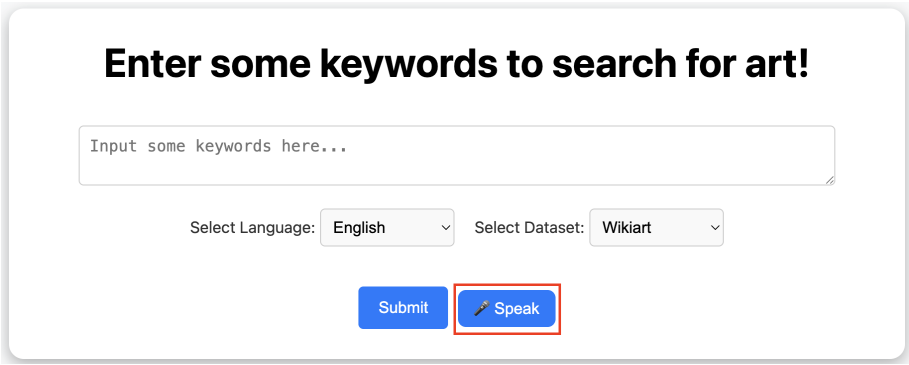

Figure 4.13: Interface – Speech-to-Text

4.8.5 Readability

Following principle 3.1 of the WCAG, the website ensures readability by avoiding jargon or unusual vocabulary. The language of each page can be programmatically determined, and the content does not require a reading level higher than lower secondary education.

Moreover, clinical research has shown that color vision deficiencies become more common with age [109]. It is therefore beneficial to include the option for users to toggle color schemes within the interface. Further studies suggest that recommended color schemes for these individuals include warm colors [110] and high-contrast options [111]. Consequently, a button was added to the application's navigation bar, allowing users to switch between these schemes. This feature is illustrated in Figure 4.14.

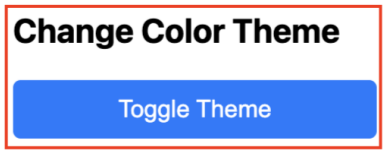

Figure 4.14: Interface – Toggle Theme

4.8.6 Understandability

While there is no specific WCAG standard that mandates providing an audio alternative to on-screen text, we determined that it would be a valuable addition to the application. As a result, we included a Text-to-Speech (TTS) button that can read aloud the contents of each webpage, as illustrated in Figure 4.15. This button is available on every page and is particularly useful for users who may have difficulty reading, enhancing their ability to

navigate the application. Furthermore, the TTS functionality is implemented entirely on the frontend, offering several advantages such as native browser support, fast performance, and automatic adaptation to the user's preferred language settings.

Figure 4.15: Interface – Text-to-Speech

Chapter 5

Results and Analysis

5.1 LMMs

This section presents the evaluation of LMMs, which are responsible for generating artwork descriptions. These descriptions are used in all system functionalities, making this step critical. We evaluate the models based on semantic similarity, runtime, and sensitivity to resolution.

5.1.1 Runtime and resource

After generating all the descriptions, it was possible to extract relevant technical information for each model. Table 5.1 presents this information, which is considered in the selection of the final model. Memory reserved and memory allocated are key limiting factors that enable (or disable) the parallelization of multiple instances of the LMM, thereby affecting the overall processing speed. The maximum pixel size is also important: higher values enable the analysis of higher-resolution images, which may improve results but also significantly increase VRAM consumption. Finally, the average processing time per image provides an estimate of the total time required when scaling to larger datasets.

Name	Parameters (B)	Avg Memory Allocated (MB)	Avg Memory Reserved (MB)	Max Pixel	Avg time
LLaVA 1.5	7	14,131.23	15,713.53	336x336	**4.22**
	13	26,260.52	28,155.96	336x336	6.52
LLaVA NeXT Vicuna	7	13,554.91	20,137.35	672x672[1]	7.69
	13	25,685.30	35,737.85	672x672[1]	13.46
LLaVA NeXT Mistral	7	14,963.57	19,195.52	672x672[1]	7.75
Llama 3.2 Vision	11	20,382.46	28,011.93	560x560	21.04
Qwen 2.5-VL	3	**7,183.34**	**7,903.92**	512x512[2]	10.84
	7	22,999.74	24,068.62	512x512[2]	10.68

Table 5.1: Summary of runtime specifications for multimodal SoTA models used.

5.1.2 Semantic similarity evaluation

Table 5.2 presents the mean, standard deviation, minimum, and maximum cosine similarity scores obtained for the top 3 highest means models across the different embeddings,

[1]The model supports flexible aspect ratios, adapting images into grids based on predefined resolutions like 336×672, 672×672 or 1080x336. Larger images are split into patches of 224×224 during processing.

[2]The model supports higher resolution and it will be discussed in the next subsection.

the full table can be seen in Annex C.1. These results provide a general overview of how semantically close the generated descriptions are to the ground truth.

Embedding	Model	Mean	Std	Min	Max
MiniLM-L6-v2	Llama-3.2-11B-Vision-Instruct	0.3737	0.1417	-0.0435	0.7474
	Qwen2.5-VL-3B-Instruct	0.3755	0.1456	-0.0533	0.8427
	Qwen2.5-VL-7B-Instruct	**0.3994**	0.1435	-0.0672	0.8191
bge-large-en-v1.5	llava-v1.6-vicuna-13b	0.6053	0.0788	0.3713	0.8718
	Llama-3.2-11B-Vision-Instruct	0.6084	0.0797	0.3699	0.8447
	Qwen2.5-VL-7B-Instruct	**0.6219**	0.0789	0.3839	0.8800
e5-large-v2	Llama-3.2-11B-Vision-Instruct	0.7708	0.0390	0.6497	0.8997
	Qwen2.5-VL-3B-Instruct	0.7700	0.0405	0.6595	0.9161
	Qwen2.5-VL-7B-Instruct	**0.7725**	0.0407	0.6603	0.9134
gte-large	llava-v1.6-vicuna-13b	0.8375	0.0349	0.7284	0.9477
	Llama-3.2-11B-Vision-Instruct	0.8372	0.0363	0.7192	0.9390
	Qwen2.5-VL-7B-Instruct	**0.8434**	0.0359	0.7415	0.9637

Table 5.2: Summary of the similarity statistics found for the different text embeddings and vision-language models.

5.1.3 Statistical significance

To assess whether the performance differences between the top two models for each embedding method were statistically significant, three tests were conducted: the paired t-test and the Wilcoxon signed-rank test. While the t-test assumes the distribution of differences to be normal, this assumption was not met in any case (as verified by the Shapiro–Wilk test). The Wilcoxon test, valid under symmetric but non-normal distributions, was applicable in all cases. The results can be seen in the Table 5.3.

Embedding	Compared Models	t-stat (p)	Wilcoxon W (p)	Mean Difference	Better Model
bge-large-en-v1.5	Qwen2.5-7B vs Llama-3.2-11B	$4.3 \cdot 10^{-14}$	$4.2 \cdot 10^{-13}$	0.0135	**Qwen2.5-7B**
all-MiniLM-L6-v2	Qwen2.5-7B vs Qwen2.5-3B	$5.5 \cdot 10^{-15}$	$1.4 \cdot 10^{-13}$	0.0240	**Qwen2.5-7B**
gte-large	Qwen2.5-7B vs LLaVA-1.6-Vicuna-13B	$1.0 \cdot 10^{-14}$	$8.3 \cdot 10^{-12}$	0.0059	**Qwen2.5-7B**
e5-large-v2	Qwen2.5-7B vs Llama-3.2-11B	0.031	0.12	0.0017	-

Table 5.3: Statistical comparison between the top two performing vision-language models for each embedding.

5.1.4 Resolution testing

To investigate the impact of input image resolution, we selected the model that achieved the best overall performance, Qwen 2.5 VL 7B, and evaluated it using various image sizes to analyze how resolution affects similarity scores. The statistical results of these evaluations, based on cosine similarity values, are summarized in Annex C.2.

Although the mean values between the top two performing image sizes for each embedding model are very close, statistical tests were conducted to determine whether these differences are significant. As shown in Table 5.4, with the exception of all-MiniLM, all p-values from both the t-tests and Wilcoxon tests are above the 5% threshold, indicating no statistically significant differences.

Embedding	Compared Sizes	t-stat (p-val)	Wilcoxon W (p-val)	Mean Difference	Better Size
bge-large-en-v1.5	512×640 vs 512×896	$7.4 \cdot 10^{-1}$	$5.4 \cdot 10^{-1}$	0.0004	-
all-MiniLM-L6-v2	512×512 vs 512×896	$4.0 \cdot 10^{-1}$	$5.8 \cdot 10^{-1}$	0.0017	-
gte-large	512×896 vs 512×640	$6.8 \cdot 10^{-1}$	$9.6 \cdot 10^{-1}$	0.0002	-
e5-large-v2	512×1024 vs 512×640	$6.2 \cdot 10^{-1}$	$8.4 \cdot 10^{-1}$	0.0003	-

Table 5.4: Statistical comparison between the top two performing image sizes for Qwen2.5-VL-7B-Instruct

5.1.5 Final consideration

As shown by the statistical tests in Table 5.3, **Qwen2.5 VL 7B** was considered statistically better than the other models in three out of the four embedding models evaluated. In the case of *e5-large-v2*, the Wilcoxon signed-rank test yielded a p-value of 0.12, above the 5% significance threshold. Nonetheless, the remaining tests presented notably small p-values, confirming Qwen 2.5 VL 7B as the best-performing model overall.

In addition to generating high-quality descriptions, Qwen 2.5 VL 7B also offers favorable processing times, as shown in Table 5.1. Depending on image size and server configuration, running multiple instances in parallel may be feasible, further accelerating the pre-processing phase.

Moreover, the tests comparing different input image sizes for the same model, shown in Table 5.4, reveal no statistically significant differences. Almost all p-values in these tests are greater than 5%, suggesting that Qwen 2.5 VL 7B maintains consistent performance regardless of image resolution within the tested range, with 512×512 adopted as the default input size for this implementation.

5.2 Embeddings

In this section we evaluate the embedding models used to transform both user text and artwork descriptions into vectors for similarity search. Since embeddings drive the retrieval accuracy, we analyze each model in terms of performance, efficiency, and resource usage.

5.2.1 Time Evaluation

As mentioned in Section 4.3.1, the embedding times were recorded and are presented in Table 5.5, as they may influence the decision on which embedding model will be adopted in this project. These times were measured using the *SemArt2000*, a set of 2,000 image-description pairs.

Team	Model	Text-GPU (s)	Text-CPU (s)	Image (s)
OpenCLIP	ViT-G/14	9.38	248.13	136.06
	ConvNeXt-XXLarge	9.23	245.58	112.08
	ConvNeXt-Large@320px	4.49	103.45	**74.85**
Google	ViT-H-14-378-quickgelu (DFN)	9.78	266.31	236.17
BAAI	bge-large-en-v1.5	22.56	623.19	N/A
SentenceTransformers	all-MiniLM-L6-v2	**2.41**	**26.55**	N/A

Team	Model	Text-GPU (s)	Text-CPU (s)	Image (s)
DAMO Academy	GTE-large	22.79	616.27	N/A
intfloat	E5-large-v2	22.95	617.51	N/A

Table 5.5: Embedding generation time per model.

It is worth noting that the CPU times refer to the embedding of 2,000 text descriptions. These times are expected to scale linearly with the number of embeddings. In contrast, the GPU times benefit from parallel computation and are less sensitive to increases in input size, making them more suitable for larger-scale processing.

5.2.2 Text-Based Retrieval Performance

Table 5.6 presents the recall results for all tested models, using the *SemArt2000* generated descriptions from the Qwen2.5 VL model as the database and the original descriptions [97] as queries.

Team	Model	Recall@1	Recall@3	Recall@6
OpenCLIP	ViT-G/14	0.084	0.146	0.201
	ConvNeXt-Large@320px	0.054	0.099	0.130
	ConvNeXt-XXLarge	0.070	0.116	0.169
Google	ViT-H-14-378-quickgelu (DFN)	0.064	0.099	0.136
BAAI	bge-large-en-v1.5	0.101	0.161	0.216
SentenceTransformers	all-MiniLM-L6-v2	0.054	0.102	0.141
DAMO Academy	**GTE-large**	**0.121**	**0.186**	**0.251**
intfloat	E5-large-v2	0.098	0.172	0.225

Table 5.6: Recall@K retrieval performance using text-only FAISS index

5.2.3 Dual-Index Retrieval Strategy

Initially, we attempted to add image embeddings into the same FAISS index alongside the text embeddings. However, since image and text embeddings have different distributions and origins, the metric barely changes as it can be shown in Table 5.7. The retrieval tended to prioritize text embeddings heavily, ignoring the contribution of the image embeddings.

Team	Model	Recall@1	Recall@3	Recall@6
OpenCLIP	ViT-G/14	**0.089**	0.148	0.206
	ConvNeXt-Large@320px	0.057	0.102	0.131
	ConvNeXt-XXLarge	0.078	0.127	0.184
Google	ViT-H-14-378-quickgelu (DFN)	0.069	0.105	0.148

Table 5.7: Results of the Recall@K results by models supporting images embeddings.

To address this and preserve the richness of the text-based and image-based information, we developed a method that considers both image and text embeddings more fairly. Instead of combining them in a single index, we created two separate FAISS indexes: one for image embeddings and another for text embeddings. During retrieval, we perform a search on both indexes and then interleave the results while ensuring there are no duplicates. Which modality will start is chosen at runtime randomly. The results can be seen in Table 5.8 and show a significant improvement in this metric between this approach and the other two mentioned.

Team	Model	Recall@1	Recall@3	Recall@6
OpenCLIP	ViT-G/14	0.188	0.357	0.462
	ConvNeXt-Large@320px	0.158	0.321	0.425
	ConvNeXt-XXLarge	0.189	0.352	0.461
Google	**ViT-H-14-378-quickgelu (DFN)**	**0.206**	**0.382**	**0.488**

Table 5.8: Recall@K results using interleaved text and image embeddings from separate FAISS indexes.

5.2.4 Subjective Evaluation by Authors

Table 5.9 shows the average number of semantically relevant images among the top-3, top-5, and top-10 results for each model considering the Table 5.8 configuration. This provides an additional perspective on model performance beyond exact matches.

Team	Model	Avg Top-3	Avg Top-5	Avg Top-10
OpenCLIP	ViT_G/14	1.67	2.33	4.67
	ConvNeXt-Large@320px	2.33	3.67	6.00
	ConvNeXt-XXLarge	1.33	2.33	5.00
Google	ViT-H-14-378-quickgelu (DFN)	1.33	2.33	4.67
BAAI	bge-large-en-v1.5	2.67	4.33	8.00
SentenceTransformers	all-MiniLM-L6-v2	2.00	3.00	4.67
DAMO Academy	**GTE-large**	**3.00**	**4.67**	**8.33**
intfloat	E5-large-v2	2.67	4.00	7.33

Table 5.9: Subjective evaluation results showing average number of relevant images retrieved in Top-K.

As observed, although the Recall@K metric suggested that models supporting image embeddings achieved significantly higher scores, a closer qualitative evaluation says differently. When analyzing three specific images, the models that relied solely on text embeddings consistently retrieved results that aligned more accurately with the intended meaning, or theme of the original descriptions. This suggests that, despite lower Recall@K values, these models might capture more effectively the semantic and contextual alignment with the original description, a dimension not fully reflected in the retrieval metrics used.

In our view, detailed text descriptions capture subtle details that image-based embeddings often miss. That's why a good text-only embedding model usually finds results that better match what we actually mean. In short, a stronger text embedding makes the search engine more accurate and useful, even if its Recall@K numbers are lower like it is in our case, because it focuses on the real meaning and context.

5.2.5 User Survey Results

The models selected for the survey were **GTE-large**, the top performer among text-only models, and **ConvNeXt-Large@320px**, the best-performing image-based model. While bge-large-en-v1.5 was the second-best model in terms of subjective evaluation, we chose ConvNeXt-Large@320px instead to contrast performance across modalities (text vs image). This choice aimed to test the hypothesis that image embeddings are more prone to hallucination than their text-based counterparts.

We collected responses from 38 participants, each evaluating 18 image pairs, corresponding to six different descriptive phrases (Box 4.3.4) with three variants each, as mentioned in Section 4.3.4. To account for individual differences in rating behavior, like conservative or extreme scorers, we applied z-normalization to each participant's responses. Table 5.10 summarizes the aggregated results per phrase, reporting the sum of the mean z-normalized scores across the three instances. Negative values indicate a preference for the first model

shown (GTE-large), while positive values favor the second (ConvNeXt). We also report the average score in cases where both top models were compared directly (*Top1 vs Top1 Avg Score*), as well as the model favored by the majority. Detailed statistics for all 18 comparisons, including means and standard deviations, are provided in Appendix D.

Phrase	Sum of Mean Z-Scores	Std Across Questions	Top1 vs Top1 Avg Score	*Favored* Model
1	-1.13	0.94	-0.52	**GTE-large**
2	-0.29	1.02	-0.13	**GTE-large**
3	0.26	1.11	0.29	ConvNeXt
4	-0.03	1.07	0.22	**GTE-large**
5	-0.54	0.89	0.13	**GTE-large**
6	0.15	0.98	-0.67	ConvNeXt

Table 5.10: Sum of mean z-normalized user ratings per phrase. Negative values favor ConvNeXt (Model 1); positive values favor GTE-large (Model 2).

5.2.6 Final consideration

Based on the combination of quantitative metrics, subjective evaluation, and user survey results shown in the last Sections, we selected the **GTE-large** model as the primary embedding model for our system. Despite not achieving the highest Recall@K scores when using multimodal retrieval strategies, GTE-large demonstrated consistent and robust performance in capturing the semantic intent of the descriptions, both in automatic and human evaluations.

Its exclusive reliance on text embeddings makes it a simpler and more scalable choice, significantly reducing the memory and computation footprint of the FAISS index, by approximately half compared to models that require both image and text embeddings. This not only streamlines the system architecture but also enhances efficiency.

Furthermore, we anticipate that future improvements in visual language models (such as Qwen 3 or Llama 4 with enhanced VL capabilities) will result in even higher-quality generated descriptions. Since our pipeline leverages text embeddings for retrieval, any gain in caption generation quality should translate directly into improved retrieval performance, without needing to modify the embedding backbone or the indexing structure.

5.3 LLMs

This section focuses on evaluating the lightweight LLMs used to generate short narratives from the selected image descriptions in the **Art Exploration** mode. We assess them based on generation time, success rate, and storytelling quality.

5.3.1 Evaluation Results

The models LLaMA 3.4 3B and Gemma 3B were excluded from testing due to consistently lower benchmark scores and reduced relevance for our use case. The selected models evaluated were **Qwen3-1.7B**, **Qwen3-4B**, and **Phi-4-mini-instruct**.

Table 5.11 summarizes the results of the manual evaluation process described in Section 4.4.1. The *Quality Score* was assigned subjectively by the authors, based on narrative coherence and how well the story incorporated all three scenes, from 1 to 5.

Model	Avg Time (s) GPU	Avg Time (s) CPU	Success Rate (%)	Avg Quality Score
Qwen3-1.7B	**12.35**	63.49	**100.0**	3.00
Qwen3-4B	17.62	141.14	90.0	**3.33**
Phi-4-mini-instruct	15.66	117.3	80.0	2.75

Table 5.11: Manual evaluation results for selected LLMs.

Some observations emerged from the evaluation:

- **Qwen3-1.7B** was the most consistent in following the prompt structure. While its outputs were not highly creative, they reliably formed coherent stories covering all descriptions.
- **Qwen3-4B** demonstrated greater narrative variation, occasionally producing higher-quality stories. However, this came with longer generation times and a slightly lower success rate, as some outputs deviated.
- **Phi-4-mini-instruct** showed the most creativity, often producing imaginative outputs. However, this led to inconsistent, with some stories focusing on only one or ignoring the descriptions entirely.

5.3.2 Final Considerations

For the current phase of the project, **Qwen3-1.7B** was selected as the preferred model. Its low generation time, high consistency in producing coherent stories, and acceptable quality scores make it a practical choice for integration into the prototype. Additionally, it requires less VRAM compared to the other evaluated models, making it more flexible for different infrastructures.

However, we noticed a recurring pattern: the model often generated stories with female main characters, regardless of the input content. This points to a possible bias in the training data, as explored in [112], which discusses how storytelling models can reinforce gender stereotypes. While this did not affect overall performance, it highlights the need to monitor and address such biases in future versions of the platform.

5.4 FAISS

Efficient similarity search is essential for real-time image retrieval in our system. FAISS handles this by indexing vector embeddings and returning the closest matches. Here, we evaluate different index types and configurations to find the best trade-off between retrieval accuracy, speed, and memory usage.

To evaluate the performance of different FAISS indexing strategies, we conducted a series of retrieval experiments using increasing dataset sizes, ranging from 1,000 to 15,000 samples, as described in Section 4.5. Three index types were tested on the CPU: IndexFlatIP, IndexIVF, and IndexIVFPQ. The number of clusters (*nlist*) and quantization parameters (*m_pq*, *nbits_pq*) were scaled appropriately with the dataset size. These parameters are required only for the second and third index types and are detailed in Appendix E.1.

5.4.1 CPU compatible

Figure 5.1 presents the evolution of Recall@1 and latency as the number of samples increases. The latency plot uses a logarithmic scale on the y-axis to better visualize differences between index types, which would otherwise be difficult to distinguish on a

linear scale. The number of samples is expressed as $\cdot 10^3$ to simplify the axis and improve readability.

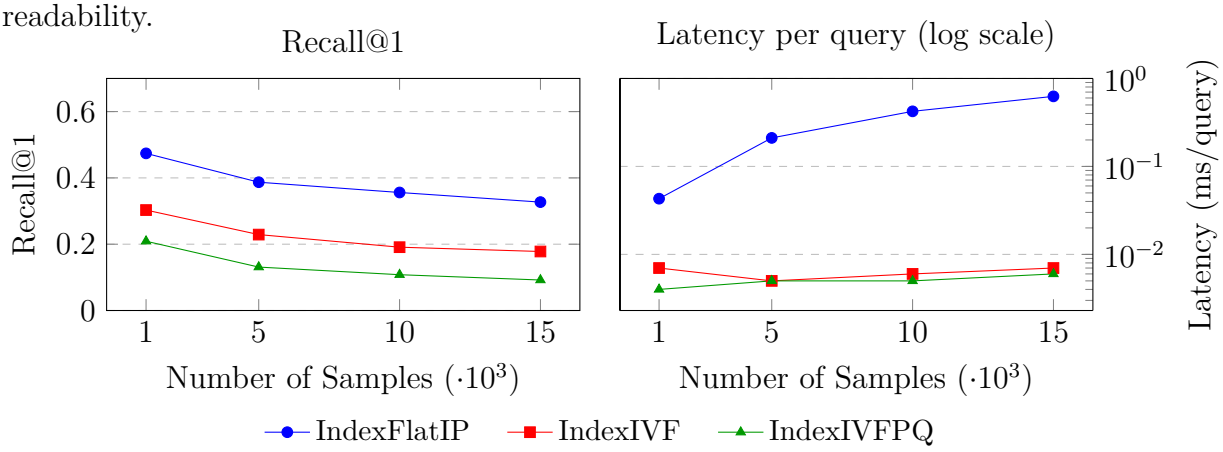

Figure 5.1: Performance of FAISS indices on CPU: (left) Recall@1 across dataset sizes; (right) Latency per query in log scale.

IndexFlatIP consistently achieves the highest recall, as it performs exact brute-force search. However, its latency increases approximately linearly with the dataset size, which appears as a curved trajectory in the log scale. In contrast, IndexIVF and IndexIVFPQ maintain significantly lower latency, mainly in larger scales, thanks to their use of clustering and compression techniques.

We observe that recall tends to decrease as the dataset size increases. This is an expected behavior: with more candidate images, the chance of confusing the correct match with a visually or semantically similar one increases. As a result, especially in top-1 evaluation, the index may return a near match rather than the exact corresponding image.

To illustrate this behavior, Figure 5.2 shows the top-6 retrieval results for a single textual query using a FAISS index with 1,000 images. The correct image is retrieved as the top-1 result, but the remaining retrieved images were considered only partially related to the query. This shows the limited diversity of a smaller dataset, which can restrict the model's ability to return multiple relevant candidates beyond the exact match.

Figure 5.2: Top-6 retrieval results using FAISS with 1,000 images. Displayed in reading order (left-to-right, top-to-bottom).

Figure 5.3 presents the same query, now using a FAISS index with 15,000 images. Although the top-1 image remains correct, the other top-6 results are more semantically aligned with the description, based on the authors' subjective evaluation. This demonstrates that increasing the dataset size enhances the richness of the retrieved set, enabling not only accurate retrieval but also more meaningful alternatives. Such behavior is particularly valuable for associative or exploratory use cases, where users benefit from seeing a broader range of conceptually relevant images.

Figure 5.3: Top-6 retrieval results using FAISS with 15,000 images.

Figure 5.4 presents the memory consumption of each index across dataset sizes. The memory usage of IndexFlatIP increases sharply in an almost quadratic proportion, reflecting the cost of storing all full-precision vectors in memory. In contrast, IndexIVF achieves lower memory usage due to coarse clustering, and IndexIVFPQ provides the most compact representation by combining clustering with compressed vector encoding.

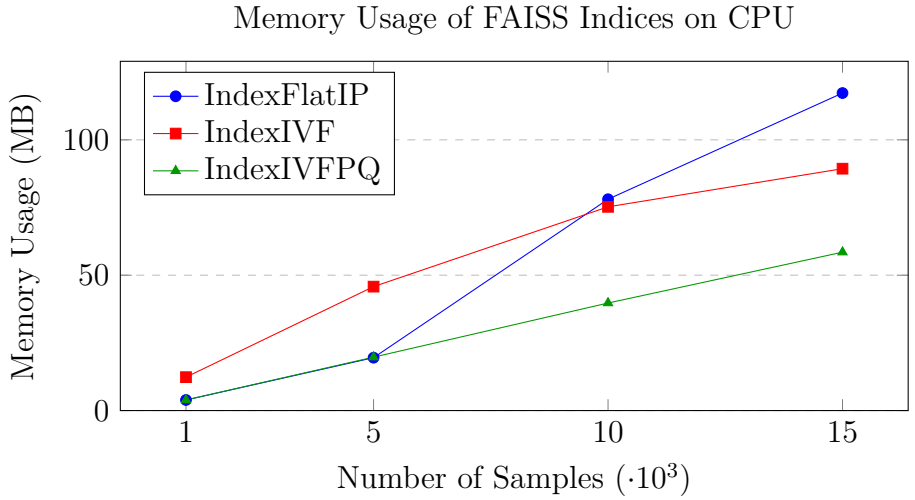

Figure 5.4: Memory usage (in MB) of different FAISS indices on CPU across dataset sizes.

5.4.2 GPU compatible

Figure 5.5 illustrates the evolution of Recall@1 and latency across increasing dataset sizes using the GPU implementation of FAISS indices. As in the CPU case, the latency plot is shown on a logarithmic scale to highlight small differences that would otherwise be imperceptible. Due to the significantly faster computation enabled by GPU acceleration, all indices exhibit extremely low query latency, even for large datasets.

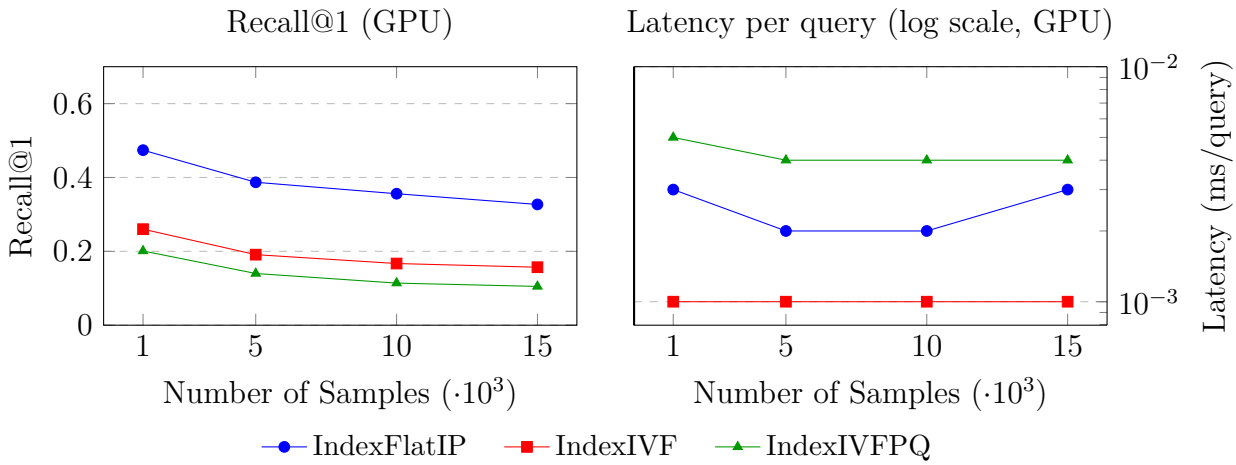

Figure 5.5: Performance of FAISS indices on GPU: (left) Recall@1 across dataset sizes; (right) Latency per query in log scale.

IndexFlatIP also achieves the highest recall, matching the CPU results, since it performs exact nearest neighbor search. While IndexIVF and IndexIVFPQ present slightly lower recall values due to their approximate nature, they offer excellent latency performance. Notably, IndexIVF's latency was so small that it fell below the measurement resolution, emphasizing the advantage of GPU parallelism for high-throughput scenarios.

Figure 5.6 shows the evolution of GPU memory consumption as the dataset size increases. As expected, IndexFlatIP demonstrates linear growth in VRAM usage due to its full storage of all vectors. IndexIVF consumes even more memory in some cases, which can be attributed to the overhead of storing the inverted lists and centroids in VRAM. In contrast, IndexIVFPQ maintains the lowest memory footprint across all dataset sizes, thanks to its use of product quantization to compress the stored vectors.

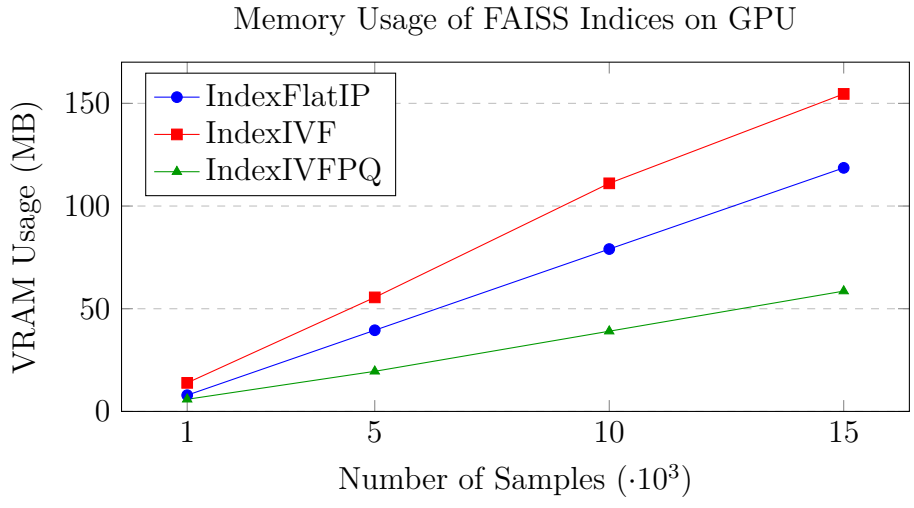

Figure 5.6: Memory usage (in MB) of different FAISS indices on GPU across dataset sizes.

5.4.3 Final Considerations

Given the characteristics of our dataset and the observed results, **IndexFlatIP** remains the most suitable choice for the platform. It achieves the highest recall across all tested configurations and maintains sub-millisecond latency even with 15,000 embeddings. Since the largest dataset considered for the platform, *WikiArtSample*, currently involves almost 52,000 candidate images, it is expected that the latency will remain within an acceptable range for interactive use. The latency growth observed in our tests appears to be nearly linear, suggesting that even with larger datasets, the increase in response time will not significantly degrade the user experience.

Moreover, if a GPU is available, the benefits are even more pronounced: IndexFlatIP delivers significantly lower latency while preserving accuracy. For future extensions involving larger-scale datasets, alternative indexing strategies such as IndexIVF or IndexIVFPQ could be considered. These methods offer lower memory consumption and faster query times at the cost of slightly reduced recall. Nevertheless, this trade-off may be acceptable if the retrieved results remain semantically relevant, which is what we are aiming for in the platform.

5.5 Text Segmentation

This section evaluates how different text segmentation strategies affect the retrieval process. We compare overlapping and non-overlapping sliding windows to measure their impact on semantic consistency and visual diversity in the results.

5.5.1 Quantitative Evaluation

As shown in Table 5.12, the average cosine similarity scores remain consistently high across all segmentation strategies, indicating that each approach is generally capable of retrieving semantically aligned images. Nevertheless, the use of overlapping segments slightly reduces the diversity of the retrieved images. This is expected, as overlapping strategies reuse partial content across multiple segments, increasing the likelihood of retrieving similar or even identical images.

Segmentation Strategy	Type	Step	Average Cosine Similarity	Total Retrieved	Diversity (Unique Images)
Paragraph	-	-	0.8492	3	3
1 Sentence	Non-overlapping	1	0.8383	15	14
2 Sentences	*Non-overlapping*	*2*	0.8432	7	7
3 Sentences	Non-overlapping	3	0.8435	5	5
2 Sentences	*Overlapping*	*1*	0.8430	14	11
3 Sentences	Overlapping	1	0.8415	13	12
3 Sentences	*Overlapping*	*2*	0.8444	7	7

Table 5.12: Retrieval performance by segmentation strategy, type, and step size.

5.5.2 Diversity Criteria

For a story composed of three paragraphs (as seen in Box 4.6.3), an ideal range for unique image outputs would be between 6 and 12. This range ensures that each part of the narrative is visually represented without excessive redundancy. In this context, we rely on the *Diversity* metric, which counts only unique images.

Based on this criterion, three strategies fall outside the acceptable range: *Paragraph-level* (3 images), *3 Sentences – Non-overlapping* (5 images), and *1 Sentence – Non-overlapping* (14 images). These are therefore excluded from further consideration.

The strategies that best balance semantic relevance and diversity are:

- *2 Sentences – Non-overlapping*
- *2 Sentences – Overlapping (step=1)*
- *3 Sentences – Overlapping (step=2)*

5.5.3 Final Consideration

Each of these configurations produced a satisfactory visual narrative when reviewed qualitatively. After filtering out duplicate images, the resulting sequences preserved coherence and reflected the storyline with varying degrees of breadth. The strategies with lower diversity offered more focused and consistent image sets, while those with higher diversity enabled broader visual interpretations.

To accommodate different user needs in the *Memory Reconstruction* mode of the platform, we propose supporting two distinct segmentation profiles:

- **Conservative mode** (*3 Sentences – Overlapping (step=2)*): prioritizes focus and visual continuity with fewer but more contextually rich segments.
- **Broader mode** (*2 Sentences – Overlapping (step=1)*): provides greater narrative detail and more visual variety, while still maintaining strong semantic alignment.

To document the results, we selected the final paragraph of the story (re-shown in Box 5.5.3) and applied both strategies.

Final Paragraph Used in Segmentation Test

When I got to the house, no one was home yet. I sat on the front step and opened the envelope I had brought. Inside was a postcard from my girlfriend at the time. She was smiling in the photo, standing in a wide field with hills in the background and flowers all around. On the back, she had written, “Wish you were here. Enjoy your vacations in your uncle’s house.”

Figure 5.7 below shows the output of the conservative strategy for the last paragraph.

Figure 5.7: Conservative mode: 3-sentence overlapping (step=2).

In contrast, Figure 5.8 below illustrates the broader strategy, with more images and a more varied interpretation:

Figure 5.8: Broader mode: 2-sentence overlapping (step = 1).

5.6 Object Detection Results

We evaluated a YOLO-based object detection model fine-tuned on our dataset. This module was initially designed to support the LMMs by identifying concrete objects within artworks, enabling more detailed and accurate descriptions. This section presents the evaluation results and explains why the technique was, at last, not considered from the final system.

5.6.1 Quantitative Evaluation

After fine-tuning YOLOv11 with our selected training subset and applying various techniques to mitigate class imbalance, the overall performance of the model remained unsatisfactory. The mean average precision (mAP) remained between 0.1 and 0.2 across different configurations, and the bounding box regression loss was consistently high, reaching values around 0.8. These results indicated that the model struggled to make meaningful predictions beyond the dominant class.

The tree map in Figure 4.9 reinforces the severity of the imbalance, showing that the overwhelming majority of labeled instances belonged to the 'person' class. Most other classes appeared fewer than ten times, severely limiting the model's ability to generalize.

5.6.2 Final Consideration

Given the poor performance and the high cost of manually expanding and balancing the dataset, especially with the need for more artworks containing non-human objects, we saw that the object detection component could not be reliably integrated into the final platform. Its exclusion allowed us to focus on other features of the system, such as semantic embedding retrieval and visual storytelling, which proved more effective for describing and associating art pieces with textual input. Additionally, during testing, the components based on large multimodal models (LMMs) and embeddings were already showing more promising results. This reinforced our decision to allocate our efforts toward these approaches instead.

5.7 Overview

This chapter presented the evaluation results of the models and strategies used in our platform. Among the vision-language models tested, **Qwen2.5 VL 7B** emerged as the

top performer, achieving higher semantic similarity scores in three out of four embedding benchmarks. It also delivered consistent results across different image resolutions, which allowed us to adopt 512×512 as the default size without sacrificing quality. Overall, this model proved to be a reliable choice for generating the image descriptions used throughout the system.

For the embedding model, we selected **GTE-large**. Although it didn't score highest in the automatic retrieval tests, it achieved the best average performance on the MTEB benchmark among the models we evaluated. It also stood out in both our subjective evaluation and the user survey, confirming its ability to capture the intended meaning of descriptions more effectively. As a text-only embedding model, GTE-large also has the advantage of simplifying our indexing pipeline while reducing memory and processing requirements.

For the **Art Exploration** mode, we tested several lightweight LLMs to generate a coherent story based on the images selected by the user. The evaluation considered average generation time and the models' ability to interpret the visual descriptions and produce a meaningful narrative. The model selected was **Qwen3 1.7B**, which consistently produced satisfactory stories while maintaining low computational cost.

In terms of FAISS indexing, **IndexFlatIP** was chosen for its superior recall and predictable latency scaling. It remained efficient even with larger datasets and offered sub-millisecond query times when used with a GPU. While other index types like IVF and IVFPQ use less memory, their reduced accuracy and added complexity made them less suitable for our current needs.

Our analysis of text segmentation strategies led us to adopt two configurations in the platform. The **conservative mode**, based on overlapping 3-sentence segments with a step of 2, provides coherent and focused visual outputs. The **broader mode**, using overlapping 2-sentence segments with a step of 1, generates more varied results. These two modes are designed to give users flexibility in how much visual detail they want when reconstructing a narrative.

Although the object detection module was initially expected to enrich each image's description by identifying specific elements, its performance was insufficient to justify integration. The YOLOv11 model showed a strong bias toward certain classes, particularly "person", due to severe class imbalance in the dataset. As a result, its predictions were not reliable, and the component was excluded from the final platform.

Together, the choices made, as shown in 5.13, represent a balanced solution, combining performance, efficiency and user adaptability, to support our goal of generating rich, meaningful visual storytelling.

Component	**Selected Model / Strategy**	**Reason for Selection**	**Notes**
Description Generation (LMM)	Qwen2.5 VL 7B	Best semantic similarity overall Consistent across resolutions	Default resolution set to 512×512
Text Embeddings	GTE-large	Best average MTEB score Strong user survey performance	Text-only model simplifies pipeline
Story Generation (LLM)	Qwen3 1.7B	Lightweight, consistent and coherent narrative generation	Used in **Art Exploration** mode
Similarity Search	IndexFlatIP (FAISS)	High recall and low latency	GPU-compatible
Text Segmentation	*Overlapping 3-sentence (step 2)* *Overlapping 2-sentence (step 1)*	Balanced between coherence and diversity	Choose between conservative and broader modes
Object Detection	No selection	Poor performance due to class imbalance	YOLOv11 discarded

Table 5.13: Summary of Evaluation Results and Final Choices

5.7.1 Example: Memory Reconstruction mode

As previously explained, this mode focuses on the user-provided story and aims to connect it with a sequence of images. The goal is to help trigger memories through the emotional and visual power of art.

For this example, we include a simple narrative to illustrate how the system works. The story shown in Box 5.7.1, is designed to evoke a past experience by describing locations, actions, and general atmosphere. In this case, it was inspired by "A Farewell to Arms" by Ernest Hemingway.

Example Story

In the late summer of that year we lived in a house in a village. It looked across the river and the plain to the mountains. In the bed of the river there were pebbles and boulders, dry and white in the sun. The water was clear and swiftly moving and blue in the channels. Troops went by the house and down the road and the dust they raised powdered the leaves of the trees. The trunks of the trees too were dusty and we saw the troops marching along the road and the dust rising. The leaves, stirred by the breeze, falling and the soldiers marching and afterward the road bare and white except for the leaves.

Once the story is provided, three settings can be adjusted: **language**, **dataset**, and **text segmentation**. The first option allows the story to be written in different languages. While this may take a bit more processing time, it ensures accessibility for diverse users. The second lets the user choose among the three available datasets used in this project: *SemArt15000*, *WikiArtSampled* and the *RoyalArchive*. The third, described in Section 5.5, refers to the strategy used to break down the story. Users can choose between a *Broader* segmentation (favoring variety and visual richness) and a more *Conservative* one (prioritizing coherence and continuity). An illustration of these configurable options is shown in Figure 5.9.

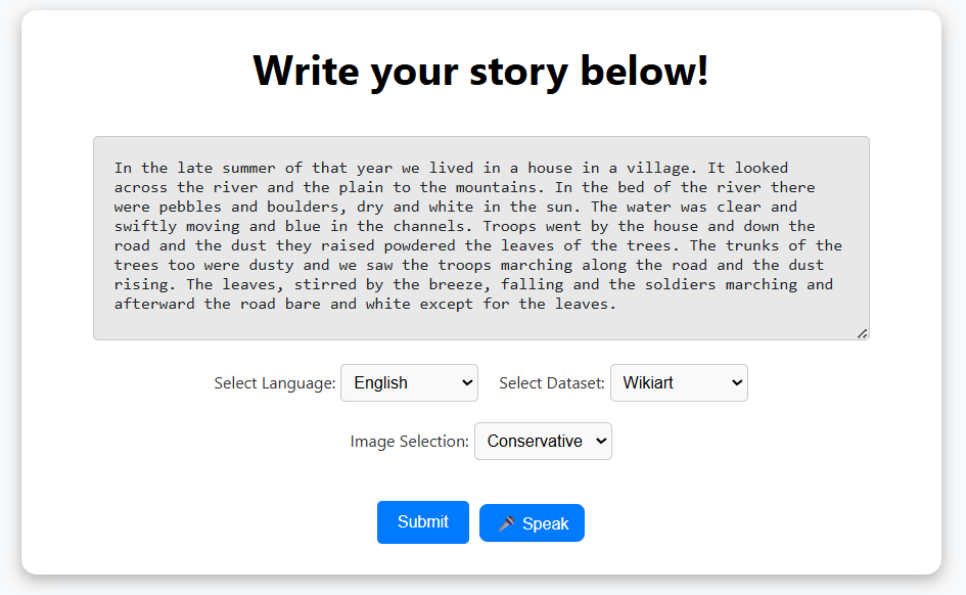

Figure 5.9: User story input and selection of settings.

After that, a sequence of text segments, also referred to as sections of the story, will be displayed alongside the top-1 retrieved image for each segment, based on the selected dataset. The user can view the image chosen by the system using the *Conservative* segmentation strategy. If the top-1 result is not satisfactory, the user can click on the

"Show 5 Images per Section" button to explore the top-5 results suggested by the retrieval engine, as illustrated in Figure 5.10.

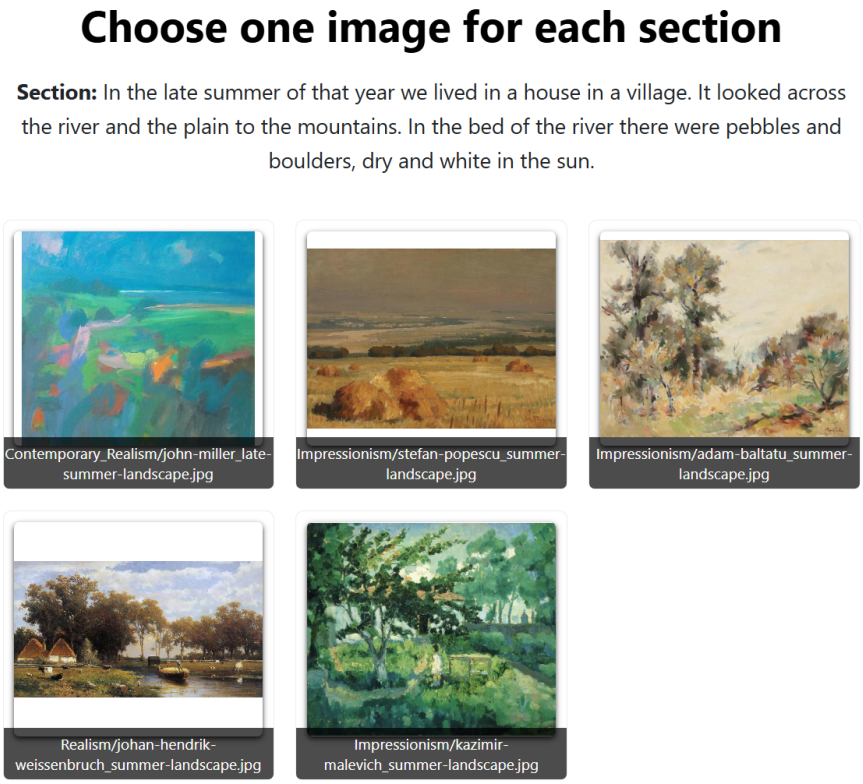

Figure 5.10: Example of a story segment using the *Conservative* segmentation.

Figure 5.11 shows a similar example using the *Broader* segmentation strategy, which emphasizes more diverse and visually rich results. Additionally, if logged in, the user has the option to save the story along with the associated set of images.

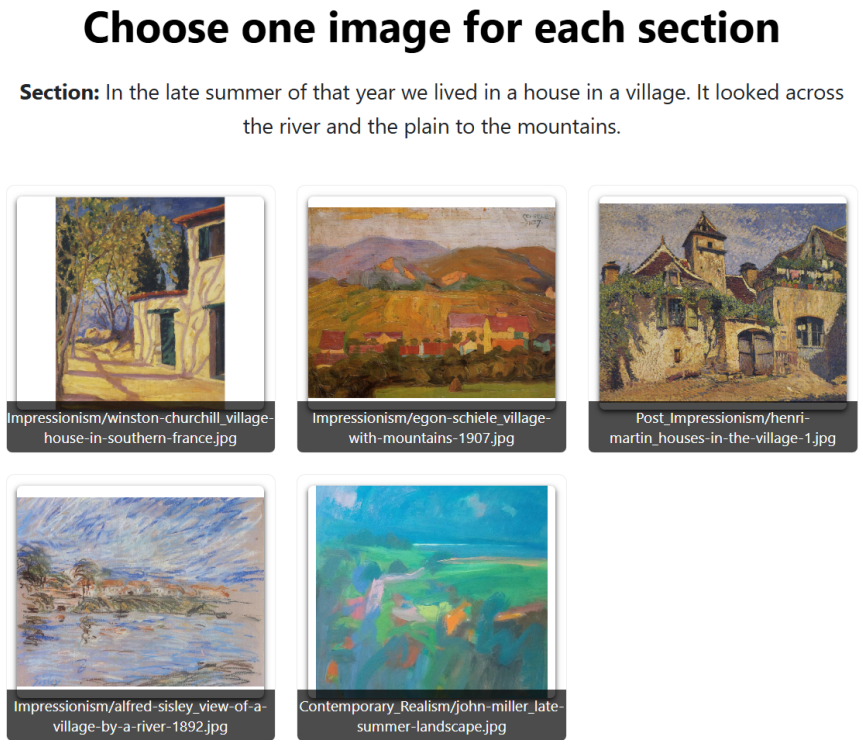

Figure 5.11: Example of a story segment using the *Broader* segmentation.

5.7.2 Example: Art Exploration Mode

In this mode, the user has the possibility of selecting images and the platform will generate a short story using their descriptions as input to the language model. This mode allows for exploratory interaction, where users can search the dataset using keywords or short texts, select images as they wish, potentially from multiple searches, and then generate a story from the selected content.

After entering a word or short phrase, the user can search the database and select one or more images. If not satisfied with the results, they can perform additional searches without losing the images already selected. As in the previous mode, both language and dataset can be changed according to user preferences.

In this example, the user searched for the following themes:

- A lonely lighthouse;
- Stormy sea;
- A small fishing boat.

After selecting three images, shown in Figure 5.12, the user can click on "Generate Story" to produce a narrative based on the selected artworks.

Figure 5.12: Images selected by the user based on the search queries.

The story generated by the model is presented in Box 5.7.2. If desired, the user can regenerate the story to explore different narrative possibilities.

Generated Story

The sun dipped below the horizon, painting the sky in hues of orange and purple as the lighthouse at dusk glowed warmly against the darkening sky. A lone fishing boat drifted gently on the sea, its silhouette barely visible in the fading light. The wind howled through the trees, carrying the scent of salt and the distant cry of a storm.

As the day ended, the boat turned back toward the shore, leaving behind a trail of waves that shimmered in the dying light. The storm had been relentless, battering the sea with relentless fury. The fishing boat had sailed through the chaos, its crew brave and steady, navigating the raging waters. But as the storm waned, the boat found shelter in a safe cove, where the calm of the day greeted it once more. The lighthouse, still standing tall, watched over the waters, a beacon of hope in the midst of the storm.

The boat's crew gathered, sharing stories and the warmth of camaraderie, as the sun began to rise again, casting a new light on the sea. The night was quiet, the sea calm, and the lighthouse stood as a symbol of resilience. The fishing boat, now resting in the cove, was ready to set out once more, its crew hopeful for the next day's adventure. The storm had passed, and with it, the promise of new beginnings. The lighthouse at dusk, the stormy sea, and the fishing boat all stood as reminders of life's unpredictable yet beautiful journey.

Chapter 6

Conclusion

6.1 Summary and Main Findings

This thesis presented the development and evaluation of **ArtEvoke**, a prototype platform that uses artificial intelligence to support cognitive stimulation in Alzheimer's patients through visual cues derived from artworks to help evoke personal memories. ArtEvoke was designed around two modes of interaction: **Memory Reconstruction**, which selects a series of artworks based on a user-provided personal story, and **Art Exploration**, which generates a story inspired by selected artworks. The evaluation of multiple models showed that the semantic similarity retrieval pipeline based on embeddings and FAISS indexing can produce relevant matches between text, in the form of stories, and images, in the form of the generated descriptions, offering promising results for therapeutic purposes.

6.2 Limitations and Improvement Areas

Several limitations should be acknowledged in this study:

- **Hardware and Software Constraints:** The system available to test and run the models was limited by its hardware. Some models that were initially of interest could not be tested due to insufficient VRAM or incompatibility with driver versions. This may have prevented the inclusion of higher-performing options.
- **Dataset Bias and Scope**: The dataset given from the Royal Museum, while a valuable and curated source of artworks, had some limitations. The total number of pieces was relatively limited, constraining the variety of matches possible during retrieval. Additionally, a predominance of works by certain artists may have introduced bias in the results.
- **Retrieval Redundancy:** Some images appeared frequently across multiple queries, likely due to disproportionately high similarity scores. This behavior may reduce the diversity of visual content presented and suggests that incorporating diversity-promoting retrieval mechanisms could improve user experience.
- **Static Text Segmentation**: The segmentation of user input stories was handled as a pre-processing step using fixed-size sliding windows. While simple and efficient, this approach does not account for variations in sentence structure or semantics, potentially causing some relevant matches to be missed.

- **Spelling and Grammar Correction**: Although basic spell-checking was included in the pre-processing pipeline, the quality of this step was not rigorously evaluated. Since grammatical and spelling errors can affect embedding generation, this may have introduced noise into the input, potentially impacting retrieval performance.
- **Translation Quality**: For inputs not originally written in English, automatic translation was applied to ensure compatibility with the models. However, the quality of these translations was not validated, and any loss of nuance or meaning may have negatively affected retrieval results, particularly in the **Memory Reconstruction** mode.

6.3 Future Work

Based on the findings and limitations of this study, several directions for future research and development are proposed:

- **Clinical Validation:** To assess the actual potential of the platform, a formal evaluation should be conducted with Alzheimer's patients. A study, supervised by qualified healthcare professionals, would help determine whether the platform contributes meaningfully to memory recall or cognitive engagement. This would also provide data for further tuning and validation of the platform in a clinical setting.
- **New Modes of Interaction:** In addition to the existing Memory Reconstruction and Art Exploration modes, future development could explore new ways of engaging users with the artworks and their descriptions. For instance, a *Memory Matching* game mode, where users try to match descriptions or stories to the correct artworks, reinforcing recognition through interaction. These new modes could offer varied cognitive stimuli and increase user engagement over time.
- **Diversification in Retrieval:** Future improvements to the retrieval system should increase the diversity and personal relevance of the results. Some images appeared more across different queries, which may reduce user engagement over time. Also, a more personalized retrieval strategy, taking into account the patient's background, such as age, place of birth, and cultural context, could help present artworks that are not only semantically relevant but also emotionally familiar. This added layer of personalization could enhance the therapeutic impact by fostering a deeper sense of recognition and connection.
- **Improved Descriptions:** Given the evolving landscape of LMMs and embedding models, future work could adopt a cascaded approach. Descriptions generated by the current best-performing model could serve as input, as well as the image itself, to newer or more powerful models to further refine and enrich the text. This iterative refinement could substantially enhance both semantic accuracy and creative expressiveness.
- **Dynamic Text Segmentation**: Implementing runtime segmentation strategies that adapt to each story's structure could improve retrieval relevance. For example, multiple segmentation schemes (with different step sizes or semantic chunking) could be tested, and the one that achieves the highest similarity score would be

selected. However, this would require more computational resources and more efficient optimization routines.

- **Enhanced Story Generation for Art Exploration Mode:** The Art Exploration mode, while functional, could benefit from more advanced prompt engineering and fine-tuned LLMs. By introducing meta-parameters and optimizing prompts based on image features, it would be possible to generate more creative, personalized, and coherent stories, while helping reduce biases such as the recurring gender pattern observed.

Addressing these areas will help make the platform more robust, flexible, and better suited for real-world use. While ArtEvoke began as a research prototype, it has potential to become a practical tool to support memory care. To get there, ongoing collaboration with healthcare professionals and the people who will actually use the platform is essential.

6.4 Platform Access

The **ArtEvoke** platform is **publicly accessible** at *www.artevoke.com.br*, offering users the opportunity to explore its features and interact with the prototype. Additionally, the full source code, test scripts, and documentation are available in the open **GitHub repository**: *ArtEvoke*. These resources are provided to encourage further exploration and improvement of the platform by researchers and developers interested in AI-assisted cognitive stimulation.

References

[1] World Health Organization. (2017) Dementia: number of people affected to triple in next 30 years. Accessed: 2025-05-11. [Online]. Available: https://www.who.int/news/item/07-12-2017-dementia-number-of-people-affected-to-triple-in-next-30-years

[2] B. Woods, H. K. Rai, E. Elliott, E. Aguirre, M. Orrell, and A. Spector, "Cognitive stimulation to improve cognitive functioning in people with dementia," *Cochrane Database of Systematic Reviews*, vol. 1, no. 1, p. CD005562, January 31 2023, epub 2023 Jan 31. [Online]. Available: https://doi.org/10.1002/14651858.CD005562.pub3

[3] World Health Organization, "Dementia: fact sheets," https://www.who.int/news-room/fact-sheets/detail/dementia, 2023, accessed: 2025-01-15.

[4] Alzheimer's Association, "2024 alzheimer's disease facts and figures," *Alzheimer's & Dementia*, vol. 20, no. 5, pp. 3708–3821, 2024, accessed: 2025-01-15. [Online]. Available: https://alz-journals.onlinelibrary.wiley.com/doi/abs/10.1002/alz.13809

[5] World Health Organization, "Dementia: Facts in pictures," https://www.who.int/news-room/facts-in-pictures/detail/dementia, 2021, accessed: 2025-01-17.

[6] Centers for Disease Control and Prevention, "About dementia," 2024, accessed: 2025-01-17. [Online]. Available: https://www.cdc.gov/alzheimers-dementia/about

[7] T. Pardo-Moreno, A. González-Acedo, A. Rivas-Domínguez, V. García-Morales, F. J. García-Cozar, J. J. Ramos-Rodríguez, and L. Melguizo-Rodríguez, "Therapeutic approach to alzheimer's disease: Current treatments and new perspectives," *Pharmaceutics*, vol. 14, no. 6, 2022. [Online]. Available: https://www.mdpi.com/1999-4923/14/6/1117

[8] J. D. Omura, L. C. McGuire, R. Patel, M. Baumgart, R. Lamb, E. M. Jeffers, B. S. Olivari, J. B. Croft, C. W. Thomas, and K. Hacker, "Modifiable risk factors for alzheimer disease and related dementias among adults aged 45 years — united states, 2019," *Morbidity and Mortality Weekly Report (MMWR)*, vol. 71, no. 20, pp. 680–685, 2022, accessed: 2025-01-17. [Online]. Available: https://www.cdc.gov/mmwr/volumes/71/wr/pdfs/mm7120a2-H.pdf

[9] Y. Sun, C.-C. Liu, C.-Y. Li, and M.-J. Chiu, "Survival after the diagnosis of mild-to-moderate alzheimer's disease dementia: A 15-year national cohort study in taiwan," *International Journal of Geriatric Psychiatry*, vol. 39, no. 9, p. e6152, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gps.6152

[10] E. B. Larson, M.-F. Shadlen, L. Wang, W. C. McCormick, J. D. Bowen, L. Teri, and W. A. Kukull, "Survival after initial diagnosis of alzheimer disease," *Annals of Internal Medicine*, vol. 140, no. 7, pp. 501–509, 2004, pMID: 15068977. [Online]. Available: https://www.acpjournals.org/doi/abs/10.7326/0003-4819-140-7-200404060-00008

[11] M. F. Folstein, S. E. Folstein, and P. R. McHugh, ""mini-mental state": A practical method for grading the cognitive state of patients for the clinician," *Journal of Psychiatric Research*, vol. 12, no. 3, pp. 189–198, 1975. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022395675900266

[12] Z. S. Nasreddine, N. A. Phillips, V. Bédirian, S. Charbonneau, V. Whitehead, I. Collin, J. L. Cummings, and H. Chertkow, "The montreal cognitive assessment, moca: A brief screening tool for mild cognitive impairment," *Journal of the American Geriatrics Society*, vol. 53, no. 4, pp. 695–699, 2005. [Online]. Available: https://agsjournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.1532-5415.2005.53221.x

[13] X. Li, M. Ji, H. Zhang, Z. Liu, Y. Chai, Q. Cheng, Y. Yang, D. Cordato, and J. Gao, "Non-drug therapies for alzheimer's disease: A review," *Neurology and Therapy*, vol. 12, no. 1, pp. 39–72, 2023. [Online]. Available: https://doi.org/10.1007/s40120-022-00416-x

[14] A. Dove, Y. Shang, W. Xu, G. Grande, E. J. Laukka, L. Fratiglioni, and A. Marseglia, "The impact of diabetes on cognitive impairment and its progression to dementia," *Alzheimer's & Dementia: The Journal of the Alzheimer's Association*, vol. 17, no. 11, pp. 1769–1778, Nov 2021. [Online]. Available: https://doi.org/10.1002/alz.12482

[15] K. Hötting and B. Röder, "Beneficial effects of physical exercise on neuroplasticity and cognition," *Neuroscience Biobehavioral Reviews*, vol. 37, no. 9, Part B, pp. 2243–2257, 2013, cNTRICS: Modeling psychosis related cognition in animal systems to enhance translational research + Life-Span Plasticity of Brain and Behavior: A Cognitive Neuroscience Perspective. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0149763413001012

[16] G. Livingston, J. Huntley, A. Sommerlad, D. Ames, C. Ballard, S. Banerjee, C. Brayne, A. Burns, J. Cohen-Mansfield, C. Cooper, S. G. Costafreda, A. Dias, N. Fox, L. N. Gitlin, R. Howard, H. C. Kales, M. Kivimäki, E. B. Larson, A. Ogunniyi, V. Orgeta, K. Ritchie, K. Rockwood, E. L. Sampson, Q. Samus, L. S. Schneider, G. Selbæk, L. Teri, and N. Mukadam, "Dementia prevention, intervention, and care: 2020 report of the *Lancet* commission," *The Lancet*, vol. 396, no. 10248, pp. 413–446, Aug. 2020. [Online]. Available: https://doi.org/10.1016/S0140-6736(20)30367-6

[17] J. D. Huntley, A. Hampshire, D. Bor, A. Owen, and R. J. Howard, "Adaptive working memory strategy training in early alzheimer's disease: randomised controlled trial," *The British Journal of Psychiatry*, vol. 210, no. 1, pp. 61–66, January 2017, epub 2016 Oct 6. [Online]. Available: https://doi.org/10.1192/bjp.bp.116.182048

[18] G. Aşiret and S. Kapucu, "The effect of reminiscence therapy on cognition, depression, and activities of daily living for patients with alzheimer disease," *Journal of geriatric psychiatry and neurology*, vol. 29, 08 2015.

[19] D. W. Zaidel, "Art and brain: insights from neuropsychology, biology and evolution," *Journal of Anatomy*, vol. 216, no. 2, pp. 177–183, February 2010. [Online]. Available: https://doi.org/10.1111/j.1469-7580.2009.01099.x

[20] D. Gollin, C. Ruaro, A. Peruzzi, E. Talassi, A. Ferrari, A. Codemo *et al.*, "Improvement of functional status in alzheimer's disease patients after cognitive activation therapy (cat)," *Non-Pharmacological Therapies in Dementia*, vol. 3, pp. 23–34, 2012.

[21] M. Campisi, L. Cannella, D. Celik, C. Gabelli, D. Gollin, M. Simoni, C. Ruaro, E. Fantinato, and S. Pavanello, "Mitigating cellular aging and enhancing cognitive functionality: visual arts-mediated cognitive activation therapy in neurocognitive disorders," *Frontiers in Aging Neuroscience*, vol. 16, 2024. [Online]. Available: https://doi.org/10.3389/fnagi.2024.1354025

[22] S. Bajpai, M. Tripathi, R. M. Pandey, A. B. Dey, and A. Nehra, "Development and validation of cognitive training intervention for alzheimer's disease (cti-ad): A picture-based interventional program," *Dementia (London, England)*, vol. 19, no. 4, pp. 1203–1219, May 2020, epub 2018 Sep 4. [Online]. Available: https://doi.org/10.1177/1471301218797043

[23] S.-T. Cheng, "Dementia caregiver burden: a research update and critical analysis," *Current Psychiatry Reports*, vol. 19, no. 9, p. 64, 2017. [Online]. Available: https://doi.org/10.1007/s11920-017-0818-2

[24] M. Nemcikova, Z. Katreniakova, and I. Nagyova, "Social support, positive caregiving experience, and caregiver burden in informal caregivers of older adults with dementia," *Frontiers in Public Health*, vol. 11, 2023. [Online]. Available: https://www.frontiersin.org/journals/public-health/articles/10.3389/fpubh.2023.1104250

[25] W. Zhang, Y. Li, W. Ren, and B. Liu, "Artificial intelligence technology in alzheimer's disease research," *Intractable Rare Diseases Research*, vol. 12, no. 4, pp. 208–212, 2023.

[26] F. Angelucci, A. R. Ai, L. Piendel, J. Cerman, and J. Hort, "Integrating ai in fighting advancing alzheimer: diagnosis, prevention, treatment, monitoring, mechanisms, and clinical trials," *Current Opinion in Structural Biology*, vol. 87, p. 102857, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959440X24000848

[27] M. S. Treder, S. Lee, and K. A. Tsvetanov, "Introduction to large language models (llms) for dementia care and research," *Frontiers in Dementia*, vol. 3, 2024. [Online]. Available: https://www.frontiersin.org/journals/dementia/articles/10.3389/frdem.2024.1385303

[28] B. A. y Arcas, "Do large language models understand us?" *Daedalus*, vol. 151, no. 2, pp. 183–197, 05 2022. [Online]. Available: https://doi.org/10.1162/daed_a_01909

[29] D. Huang, C. Yan, Q. Li, and X. Peng, "From large language models to large multimodal models: A literature review," *Applied Sciences*, vol. 14, no. 12, 2024. [Online]. Available: https://www.mdpi.com/2076-3417/14/12/5068

[30] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[31] A. F. Agarap, "Deep learning using rectified linear units (relu)," 2019. [Online]. Available: https://arxiv.org/abs/1803.08375

[32] M. Gabrié, S. Ganguli, C. Lucibello, and R. Zecchina, "Neural networks: from the perceptron to deep nets," 2023. [Online]. Available: https://arxiv.org/abs/2304.06636

[33] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014. [Online]. Available: https://arxiv.org/abs/1412.3555

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[35] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278919305974

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[37] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, p. 36–45, Jan. 1966. [Online]. Available: https://doi.org/10.1145/365153.365168

[38] Supriyono, A. P. Wibawa, Suyono, and F. Kurniawan, "Advancements in natural language processing: Implications, challenges, and future directions," *Telematics and Informatics Reports*, vol. 16, p. 100173, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772503024000598

[39] V. Zouhar, C. Meister, J. L. Gastaldi, L. Du, T. Vieira, M. Sachan, and R. Cotterell, "A formal perspective on byte-pair encoding," 2024. [Online]. Available: https://arxiv.org/abs/2306.16837

[40] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. [Online]. Available: https://arxiv.org/abs/1810.04805

[41] S. J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Gallé, A. Raja, C. Si, W. Y. Lee, B. Sagot, and S. Tan, "Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp," 2021. [Online]. Available: https://arxiv.org/abs/2112.10508

[42] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2023. [Online]. Available: https://arxiv.org/abs/1910.10683

[43] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[44] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019. [Online]. Available: https://arxiv.org/abs/1910.13461

[45] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, P. Payne, M. Seneviratne, P. Gamble, C. Kelly, A. Babiker, N. Schärli, A. Chowdhery, P. Mansfield, D. Demner-Fushman, B. A. y Arcas, D. Webster, G. S. Corrado, Y. Matias, K. Chou, J. Gottweis, N. Tomasev, Y. Liu, A. Rajkomar, J. Barral, C. Semturs, A. Karthikesalingam, and V. Natarajan, "Large language models encode clinical knowledge," *Nature*, vol. 620, no. 7972, pp. 172–180, 2023. [Online]. Available: https://doi.org/10.1038/s41586-023-06291-2

[46] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021. [Online]. Available: https://arxiv.org/abs/2010.11929

[47] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021. [Online]. Available: https://arxiv.org/abs/2103.00020

[48] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible scaling laws for contrastive language-image learning," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2023, p. 2818–2829. [Online]. Available: http://dx.doi.org/10.1109/CVPR52729.2023.00276

[49] H. Chen, X. Wang, Y. Zhou, B. Huang, Y. Zhang, W. Feng, H. Chen, Z. Zhang, S. Tang, and W. Zhu, "Multi-modal generative ai: Multi-modal llm, diffusion and beyond," 2024. [Online]. Available: https://arxiv.org/abs/2409.14993

[50] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," 2023. [Online]. Available: https://arxiv.org/abs/2301.12597

[51] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: a visual language model for few-shot learning," 2022. [Online]. Available: https://arxiv.org/abs/2204.14198

[52] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

[53] J. Li, W. Lu, H. Fei, M. Luo, M. Dai, M. Xia, Y. Jin, Z. Gan, D. Qi, C. Fu, Y. Tai, W. Yang, Y. Wang, and C. Wang, "A survey on benchmarks of multimodal large language models," 2024. [Online]. Available: https://arxiv.org/abs/2408.08632

[54] C. Xu, S. Guan, D. Greene, and M.-T. Kechadi, "Benchmark data contamination of large language models: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2406.04244

[55] R. Patil, S. Boit, V. Gudivada, and J. Nandigam, "A survey of text representation and embedding techniques in nlp," *IEEE Access*, vol. 11, pp. 36 120–36 146, 2023.

[56] AWS, "What are embeddings in machine learning?" accessed: April 9, 2025. [Online]. Available: https://aws.amazon.com/what-is/embeddings-in-machine-learning/

[57] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[58] OpenAI, "Introducing gpt-4o: Openai's most capable multimodal model," 2024, accessed: 2025-04-07. [Online]. Available: https://openai.com/index/hello-gpt-4o/

[59] S. Kakarla, G. S. B. Venkata, and A. Gaddam, "How does a multilingual lm handle multiple languages?" in *Proceedings of the Workshop on Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP)*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 435–445. [Online]. Available: https://aclanthology.org/2023.blackboxnlp-1.39

[60] S. Xu, "A guide to open-source embedding models," *BentoML Blog*, Mar 2025. [Online]. Available: https://www.bentoml.com/blog/a-guide-to-open-source-embedding-models

[61] R. Merchant, X. Li, and J. Hockenmaier, "Grammatical information in bert sentence embeddings as two-dimensional arrays," *arXiv preprint arXiv:2010.00793*, 2020.

[62] D. Naber *et al.*, "language_tool_python: A python interface to languagetool." [Online]. Available: https://pypi.org/project/language_tool-python/

[63] J. Webber, "pyspellchecker: A pure python spell checker based on peter norvig's blog post." [Online]. Available: https://pypi.org/project/pyspellchecker/

[64] G. Melis, C. Dyer, and P. Blunsom, "On the state of the art of evaluation in neural language models," *arXiv preprint arXiv:1707.05589*, 2017, [Submitted on 18 Jul 2017 (v1), last revised 20 Nov 2017 (this version, v2)].

[65] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," 2017. [Online]. Available: https://arxiv.org/abs/1702.08734

[66] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The faiss library," 2025. [Online]. Available: https://arxiv.org/abs/2401.08281

[67] Facebook AI Research, "Faiss wiki," https://github.com/facebookresearch/faiss/wiki, 2024, accessed: 2025-05-15.

[68] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.

[69] A. B. Amjoud and M. Amrouch, "Object detection using deep learning, cnns and vision transformers: A review," *IEEE Access*, vol. 11, pp. 35 479–35 516, 2023.

[70] K. Wołk and M. S. Tatara, "A review of semantic segmentation and instance segmentation techniques in forestry using lidar and imagery data," *Electronics*, vol. 13, no. 20, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/20/4139

[71] Emami, "Semantic segmentation vs instance segmentation," 2021, accessed: 2025-04-09. [Online]. Available: https://medium.com/@emamimehr/semantic-segmentation-vs-instance-segmentation-7b2f2ee1e091

[72] R. Girshick, "Fast r-cnn," 2015. [Online]. Available: https://arxiv.org/abs/1504.08083

[73] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016. [Online]. Available: https://arxiv.org/abs/1506.02640

[74] R. Khanam and M. Hussain, "Yolov11: An overview of the key architectural enhancements," 2024. [Online]. Available: https://arxiv.org/abs/2410.17725

[75] X. Yue, Y. Ni, K. Zhang, T. Zheng, R. Liu, G. Zhang, S. Stevens, D. Jiang, W. Ren, Y. Sun, C. Wei, B. Yu, R. Yuan, R. Sun, M. Yin, B. Zheng, Z. Yang, Y. Liu, W. Huang, H. Sun, Y. Su, and W. Chen, "Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi," 2024. [Online]. Available: https://arxiv.org/abs/2311.16502

[76] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[77] P. Lu, H. Bansal, T. Xia, J. Liu, C. Li, H. Hajishirzi, H. Cheng, K.-W. Chang, M. Galley, and J. Gao, "Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts," *arXiv preprint arXiv:2310.02255*, 2023. [Online]. Available: https://arxiv.org/abs/2310.02255

[78] N. Muennighoff, N. Tazi, L. Magne, and N. Reimers, "Mteb: Massive text embedding benchmark," 2023. [Online]. Available: https://arxiv.org/abs/2210.07316

[79] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, ser. 2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009. United States: IEEE Computer Society, 2009, pp. 248–255, publisher Copyright: © 2009 IEEE.; 2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009 ; Conference date: 20-06-2009 Through 25-06-2009.

[80] Y. Wang, X. Ma, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. Ren, A. Arulraj, X. He, Z. Jiang, T. Li, M. Ku, K. Wang, A. Zhuang, R. Fan, X. Yue, and W. Chen, "Mmlu-pro: A more robust and challenging multi-task language understanding benchmark," 2024. [Online]. Available: https://arxiv.org/abs/2406.01574

[81] S. Lu, Y. Li, Q.-G. Chen, Z. Xu, W. Luo, K. Zhang, and H.-J. Ye, "Ovis: Structural embedding alignment for multimodal large language model," *arXiv:2405.20797*, 2024.

[82] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," 2023.

[83] H. Liu, C. Li, Y. Li, B. Li, Y. Zhang, S. Shen, and Y. J. Lee, "Llava-next: Improved reasoning, ocr, and world knowledge," January 2024. [Online]. Available: https://llava-vl.github.io/blog/2024-01-30-llava-next/

[84] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, H. Zhong, Y. Zhu, M. Yang, Z. Li, J. Wan, P. Wang, W. Ding, Z. Fu, Y. Xu, J. Ye, X. Zhang, T. Xie, Z. Cheng, H. Zhang, Z. Yang, H. Xu, and J. Lin, "Qwen2.5-vl technical report," 2025. [Online]. Available: https://arxiv.org/abs/2502.13923

[85] Meta AI, "Introducing llama 3.2 with vision capabilities," 2024, accessed: 2025-04-07. [Online]. Available: https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/

[86] Meta AI , "Llama 4: Multimodal intelligence at scale," 2024, accessed: 2025-04-07. [Online]. Available: https://ai.meta.com/blog/llama-4-multimodal-intelligence/

[87] Z. Liu, C. Zhao, I. Fedorov, B. Soran, D. Choudhary, R. Krishnamoorthi, V. Chandra, Y. Tian, and T. Blankevoort, "Spinquant: Llm quantization with learned rotations," 2025. [Online]. Available: https://arxiv.org/abs/2405.16406

[88] G. Ilharco, M. Wortsman, R. Wightman, C. Gordon, N. Carlini, R. Taori, A. Dave, V. Shankar, H. Namkoong, J. Miller, H. Hajishirzi, A. Farhadi, and L. Schmidt, "Openclip," Jul. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5143773

[89] C. Schuhmann, R. Beaumont, R. Vencu, C. W. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, P. Schramowski, S. R. Kundurthy, K. Crowson, L. Schmidt, R. Kaczmarczyk, and J. Jitsev, "LAION-5b: An open large-scale dataset for training next generation image-text models," in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. [Online]. Available: https://openreview.net/forum?id=M3Y74vmsMcY

[90] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," 2024. [Online]. Available: https://arxiv.org/abs/2402.03216

[91] Z. Li, X. Zhang, Y. Zhang, D. Long, P. Xie, and M. Zhang, "Towards general text embeddings with multi-stage contrastive learning," 2023. [Online]. Available: https://arxiv.org/abs/2308.03281

[92] L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei, "Text embeddings by weakly-supervised contrastive pre-training," 2024. [Online]. Available: https://arxiv.org/abs/2212.03533

[93] Hugging Face, "Mteb leaderboard," https://huggingface.co/spaces/mteb/leaderboard, accessed: 2025-05-20.

[94] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, C. Zheng, D. Liu, F. Zhou, F. Huang, F. Hu, H. Ge, H. Wei, H. Lin, J. Tang, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Zhou, J. Lin, K. Dang, K. Bao, K. Yang, L. Yu, L. Deng, M. Li, M. Xue, M. Li, P. Zhang, P. Wang, Q. Zhu, R. Men, R. Gao, S. Liu, S. Luo, T. Li, T. Tang, W. Yin, X. Ren, X. Wang, X. Zhang, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Zhang, Y. Wan, Y. Liu, Z. Wang, Z. Cui, Z. Zhang, Z. Zhou, and Z. Qiu, "Qwen3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2505.09388

[95] Microsoft, :, A. Abouelenin, A. Ashfaq, A. Atkinson, H. Awadalla, N. Bach, J. Bao, A. Benhaim, M. Cai, V. Chaudhary, C. Chen, D. Chen, D. Chen, J. Chen, W. Chen, Y.-C. Chen, Y. ling Chen, Q. Dai, X. Dai, R. Fan, M. Gao, M. Gao, A. Garg, A. Goswami, J. Hao, A. Hendy, Y. Hu, X. Jin, M. Khademi, D. Kim, Y. J. Kim, G. Lee, J. Li, Y. Li, C. Liang, X. Lin, Z. Lin, M. Liu, Y. Liu, G. Lopez, C. Luo, P. Madan, V. Mazalov, A. Mitra, A. Mousavi, A. Nguyen, J. Pan, D. Perez-Becker, J. Platin, T. Portet, K. Qiu, B. Ren, L. Ren, S. Roy, N. Shang, Y. Shen, S. Singhal, S. Som, X. Song, T. Sych, P. Vaddamanu, S. Wang, Y. Wang, Z. Wang, H. Wu, H. Xu, W. Xu, Y. Yang, Z. Yang, D. Yu, I. Zabir, J. Zhang, L. L. Zhang, Y. Zhang, and X. Zhou, "Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras," 2025. [Online]. Available: https://arxiv.org/abs/2503.01743

[96] G. Team, A. Kamath, J. Ferret, S. Pathak, N. Vieillard, R. Merhej, S. Perrin, T. Matejovicova, A. Ramé, M. Rivière, L. Rouillard, T. Mesnard, G. Cideron, J. bastien Grill, S. Ramos, E. Yvinec, M. Casbon, E. Pot, I. Penchev, G. Liu, F. Visin, K. Kenealy, L. Beyer, X. Zhai, A. Tsitsulin, R. Busa-Fekete, A. Feng, N. Sachdeva, B. Coleman, Y. Gao, B. Mustafa, I. Barr, E. Parisotto, D. Tian, M. Eyal, C. Cherry, J.-T. Peter, D. Sinopalnikov, S. Bhupatiraju, R. Agarwal, M. Kazemi, D. Malkin, R. Kumar, D. Vilar, I. Brusilovsky, J. Luo, A. Steiner, A. Friesen, A. Sharma, A. Sharma, A. M. Gilady, A. Goedeckemeyer, A. Saade, A. Feng, A. Kolesnikov, A. Bendebury, A. Abdagic, A. Vadi, A. György, A. S. Pinto, A. Das, A. Bapna, A. Miech, A. Yang, A. Paterson, A. Shenoy, A. Chakrabarti, B. Piot, B. Wu, B. Shahriari, B. Petrini, C. Chen, C. L. Lan, C. A. Choquette-Choo, C. Carey, C. Brick, D. Deutsch, D. Eisenbud, D. Cattle, D. Cheng, D. Paparas, D. S. Sreepathihalli, D. Reid, D. Tran, D. Zelle, E. Noland, E. Huizenga, E. Kharitonov, F. Liu, G. Amirkhanyan, G. Cameron, H. Hashemi, H. Klimczak-Plucińska, H. Singh, H. Mehta, H. T. Lehri, H. Hazimeh, I. Ballantyne, I. Szpektor, I. Nardini, J. Pouget-Abadie, J. Chan, J. Stanton, J. Wieting, J. Lai, J. Orbay, J. Fernandez, J. Newlan, J. yeong Ji, J. Singh, K. Black, K. Yu, K. Hui, K. Vodrahalli, K. Greff, L. Qiu, M. Valentine, M. Coelho, M. Ritter, M. Hoffman, M. Watson, M. Chaturvedi, M. Moynihan, M. Ma, N. Babar, N. Noy, N. Byrd, N. Roy, N. Momchev, N. Chauhan, N. Sachdeva, O. Bunyan, P. Botarda, P. Caron, P. K. Rubenstein, P. Culliton, P. Schmid, P. G. Sessa, P. Xu, P. Stanczyk, P. Tafti, R. Shivanna, R. Wu, R. Pan, R. Rokni, R. Willoughby, R. Vallu, R. Mullins, S. Jerome, S. Smoot, S. Girgin, S. Iqbal, S. Reddy, S. Sheth, S. Põder, S. Bhatnagar, S. R. Panyam, S. Eiger, S. Zhang, T. Liu, T. Yacovone, T. Liechty,

U. Kalra, U. Evci, V. Misra, V. Roseberry, V. Feinberg, V. Kolesnikov, W. Han, W. Kwon, X. Chen, Y. Chow, Y. Zhu, Z. Wei, Z. Egyed, V. Cotruta, M. Giang, P. Kirk, A. Rao, K. Black, N. Babar, J. Lo, E. Moreira, L. G. Martins, O. Sanseviero, L. Gonzalez, Z. Gleicher, T. Warkentin, V. Mirrokni, E. Senter, E. Collins, J. Barral, Z. Ghahramani, R. Hadsell, Y. Matias, D. Sculley, S. Petrov, N. Fiedel, N. Shazeer, O. Vinyals, J. Dean, D. Hassabis, K. Kavukcuoglu, C. Farabet, E. Buchatskaya, J.-B. Alayrac, R. Anil, Dmitry, Lepikhin, S. Borgeaud, O. Bachem, A. Joulin, A. Andreev, C. Hardin, R. Dadashi, and L. Hussenot, "Gemma 3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2503.19786

[97] N. Garcia and G. Vogiatzis, "How to read paintings: Semantic art understanding with multi-modal retrieval," 2018. [Online]. Available: https://arxiv.org/abs/1810.09617

[98] WikiArt, "Wikiart.org - visual art encyclopedia," 2024, accessed: 2025-05-16. [Online]. Available: https://www.wikiart.org/en/

[99] W. R. Tan, C. S. Chan, H. Aguirre, and K. Tanaka, "Improved artgan for conditional synthesis of natural image and artwork," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 394–409, 2019. [Online]. Available: https://doi.org/10.1109/TIP.2018.2866698

[100] LanguageTool Contributors, "Languagetool: Open source grammar and style checker," https://languagetool.org/, 2024, accessed: 2025-05-17.

[101] P. Omelianchuk, A. Lagutina, A. Laposhko, E. Artemova, and A. Fenogenova, "Grammatical error correction: A survey with deep learning perspective," *arXiv preprint arXiv:2101.09362*, 2021. [Online]. Available: https://arxiv.org/abs/2101.09362

[102] N. Baccouri, "Deep translator: A flexible translation library for python," https://deep-translator.readthedocs.io/, 2020, accessed: 2025-05-17.

[103] Ultralytics, *Ultralytics YOLO Documentation*, 2024, accessed: 2025-03-02. [Online]. Available: https://docs.ultralytics.com/

[104] B. Kellenberger, D. Marcos, S. Lobry, and D. Tuia, "Half a percent of labels is enough: Efficient animal detection in uav imagery using deep cnns and active learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 12, pp. 9524–9533, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8782109

[105] MongoDB Inc., "Mongodb documentation," 2025, accessed: 2025-05-31. [Online]. Available: https://www.mongodb.com/docs/

[106] S. Ramírez, "Fastapi," https://github.com/fastapi/fastapi, 2018, accessed: 2025-05-31.

[107] Meta Platforms, Inc., *React – A JavaScript library for building user interfaces*, 2025, accessed: 2025-05-31. [Online]. Available: https://react.dev/

[108] W3C, "Web Content Accessibility Guidelines (WCAG) 2.1," https://www.w3.org/TR/WCAG21/, 2018, accessed: 2025-04-24.

[109] Wolters Kluwer Health: Lippincott Williams & Wilkins. (2014, Feb.) Color vision problems become more common with age, study shows. Accessed: 2025-05-12. [Online]. Available: https://www.sciencedaily.com/releases/2014/02/140220102614.htm

[110] M. Hencová and V. Kotradyová, "Colour in the environment for older adults," *Architecture Papers of the Faculty of Architecture and Design STU*, vol. 28, no. 4, pp. 15–23, 2023. [Online]. Available: https://doi.org/10.2478/alfa-2023-0021

[111] Z.-Y. Wang and J. Y. Cho, "Older adults' response to color visibility in indoor residential environment using eye-tracking technology," *Sensors*, vol. 22, no. 22, p. 8766, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/22/8766

[112] E. Chen, R.-J. Zhan, Y.-B. Lin, and H.-H. Chen, "From structured prompts to open narratives: Measuring gender bias in llms through open-ended storytelling," 2025. [Online]. Available: https://arxiv.org/abs/2503.15904

[113] Anthropic, "Claude 3.5 haiku," 2024, accessed: 2025-04-07. [Online]. Available: https://www.anthropic.com/claude/haiku

[114] Anthropic , "Claude 3.7 sonnet announcement," 2024, accessed: 2025-04-07. [Online]. Available: https://www.anthropic.com/news/claude-3-7-sonnet

[115] Google, "Gemini 2 family expands with flash and pro models," 2025, accessed: 2025-04-07. [Online]. Available: https://developers.googleblog.com/en/gemini-2-family-expands/

[116] P. Zhang, Z. Liu, S. Xiao, N. Shao, Q. Ye, and Z. Dou, "Long context compression with activation beacon," 2024. [Online]. Available: https://arxiv.org/abs/2401.03462

Appendix A

Models Overview and Benchmarks

The following tables compile data from official model documentation and benchmark papers. A dash (–) indicates that no official result was found.

A.1 LMM Model Specifications

Team	Name	Parameters (B)	Release Date	Context tokens (K)	Max output tokens	Weights Access
AIDC-AI	Ovis2	8	03/2025	32	Not specified	Open-source
	Ovis2	16	03/2025			
	Ovis2	34	03/2025			
Anthropic	Claude 3.5 Haiku	-	07/2024	200	8192	Closed
	Claude 3.5 Sonnet	-	04/2024		8192	
	Claude 3.7 Sonnet	-	11/2024		8.192 / 64.000	
Google	Gemini 2.0 Flash-Lite	-	02/2025	1.048	8.192	Closed
	Gemini 2.0 Flash	-	02/2025	1.048	8.192	
	Gemini 2.0 Flash Thinking	-	02/2025	1.048	65.536	
	Gemini 2.0 Pro	-	02/2025	2.097	8.192	
LLaVA	1.5	7	09/2023	4.096	Not specified	Open-source
	1.5	13	09/2023			
	NeXT Vicuna	7	01/2024			
	NeXT Vicuna	13	12/2023			
	NeXT Mistral	7	12/2023			
	NeXT	34	01/2024			
Llama	3.2 Vision	11	09/2024	128	2.048	Restricted
	3.2 Vision	90	09/2024	128	2.048	
	4 Scout	∼109	04/2025	10.000	Not specified	
	4 Maverick	∼400	04/2025	10.000	Not specified	
	4 Behemoth	2.000	04/2025	10.000	Not specified	
OpenAI	GPT4.5 Preview	-	02/2025	128	16.384	Closed
	GPT-4o	-	08/2024			
Qwen	2.5-VL	3	02/2025	32	8.192	Open-source
		7	02/2025			
		32	03/2025			

Table A.1: Overview of evaluated multimodal SoTA models with support for image inputs. Source: [58, 81–86, 113–115]

A.2 Benchmark Results

Team	Name	Parameters (B)	MMMU (0-shot)	VQAv2	MathVista
AIDC-AI	Ovis2	8	59%	-	71.40%
	Ovis2	16	59.60%	-	74.90%
	Ovis2	34	65.60%	-	77%

Team	Name	Parameters (B)	MMMU (0-shot)	VQAv2	MathVista
Anthropic	Claude 3.5 Haiku	-	60.50%	-	61.60%
	Claude 3.5 Sonnet	-	70.40%	-	65.40%
	Claude 3.7 Sonnet	-	71.60%	-	-
Google	Gemini 2.0 Flash-Lite	-	68%	-	57.60%
	Gemini 2.0 Flash	-	71.70%	-	-
	Gemini 2.0 Flash Thinking	-	75.40%	-	-
	Gemini 2.0 Pro	-	72.70%	-	-
LLaVA	1.5	7	-	78.50%	-
	1.5	13	36.40%	80%	-
	NeXT Vicuna	7	35.80%	81.80%	34.60%
	NeXT Vicuna	13	36.20%	82.80%	35.60%
	NeXT Mistral	7	35.30%	82.20%	37.70%
	NeXT	34	51.10%	83.70%	46.50%
Llama	3.2 Vision	11	50.70%	75.20%	51.50%
	3.2 Vision	90	60.30%	78.10%	57.30%
	4 Scout	∼109	69.40%	-	70.70%
	4 Maverick	∼400	73.40%	-	73.70%
	4 Behemoth	-	76.10%	-	-
OpenAI	GPT4.5 Preview	-	74.40%	-	-
	GPT-4o	-	69.10%	-	63.80%
Qwen	2.5-VL	3	53.10%	-	62.30%
	2.5-VL	7	58.60%	-	68.20%
	2.5-VL	32	70%	-	74.70%

Table A.2: Performance of multimodal models on benchmark. Results are reported for MMMU (zero-shot), VQAv2 (visual question answering), and MathVista (math + vision tasks). Source: [58, 81–87, 113–115].

A.3 Embedding Models

Team	Name	Image size	Embedding Dimension	Parameters (M)	Release Date	Modality	Weights Available
OpenCLIP	ViT-B/32	256px	512	151	Jan 2021	Image-Text	Open-source
	ViT-B/16	224px	512	150	Jul 2021		
	ViT-L/14	224px	768	428	Jan 2022		
	ViT-L/14	336px	768	428	April 2022		
	ViT-G/14	224px	1024	601	July 2023		
	ConvNeXt-XXLarge	224px	1024	848	Jan 2023		
	ConvNeXt-Large@320px	320px	1024	690	Jan 2023		
Google	ViT-SO400M/14 (SigLIP)	224px	768	632	Aug 2023	Image-Text	Open-source
	ViT-L/14 (DFN)	224px	768	428			
	ViT-SO400M-14 (SigLIP)	384px	768	632			
	ViT-H/14-quickgelu (DFN)	224px	1024	632			
	ViT-H-14-378-quickgelu (DFN)	378px	1024	632			
BAAI	BGE-M3	-	1024	569	Jan 2024	Text	Restrictive
	bge-large-en-v1.5	-	1024	335			
	bge-base-en-v1.5	-	768	110			
Sentence Transformers	all-MiniLM-L6-v2	-	384	22,7	Apr 2019	Text	Restrictive
DAMO Academy	GTE-large	-	1024	335	Aug 2023	Text	Open-source
intfloat	E5-large-v2	-	1024	335	Dec 2022	Text	Open-source

Table A.3: Overview of evaluated embeddings models. Source: [47, 88–92, 116]

A.4 Embeddings Benchmark

Team	Name	MTEB (Average)	ImageNet (Zero-shot)
OpenCLIP	ViT-B/32	–	72.8%
	ViT-B/16	–	73.5%
	ViT-L/14 - 224	–	75.3%
	ViT-L/14 - 336	–	79.2%
	ViT-G/14	–	80.1%

Team	Name	MTEB (Average)	ImageNet (Zero-shot)
	ConvNeXt-XXLarge	–	79.5%
	ConvNeXt-Large@320px	–	76.9%
Google	ViT-SO400M/14 (SigLIP)	–	82.0%
	ViT-L/14 (DFN)	–	82.2%
	ViT-SO400M-14 (SigLIP)	–	83.1%
	ViT-H/14-quickgelu (DFN)	–	83.4%
	ViT-H-14-378-quickgelu (DFN)	–	84.4%
BAAI	BGE-M3	59.56%	N/A
	bge-large-en-v1.5	45.08%	N/A
	bge-base-en-v1.5	–	N/A
Sentence Transformers	all-MiniLM-L6-v2	56.26%	N/A
DAMO Academy	GTE-large	63.13%	N/A
intfloat	E5-large-v2	62.25%	N/A

Table A.4: List of evaluated embedding models and their benchmark scores. Source: [88, 93]

A.5 LLM Models Specifications

Team	Name	Parameters (B)	Release Date	Context tokens (K)	Max output tokens	Weights Access
Qwen	3	0.6 1.7 4	04/2024	32	8.192	Open-source
Microsoft	Phi-4-mini	3.8	02/2025	200	Not specified	Open-source
Google	Gemma 3	1 4	03/2025	32 128	8192	Open-source
Meta	Llama 3.2	1 3	09/2024	128	Not specified	Open-source

Table A.5: Lightweight LLMs (<4B parameters) for story generation from text-only input. Source: [87, 94–96]

A.6 LLM Models Benchmarks

Team	Name	Parameters (B)	MMLU-Pro
Qwen	3	0.6	24.74
		1.7	36.76
		4	50.58
Microsoft	Phi-4-mini	3.8	52.8
Google	Gemma 3	1	14.7
		4	43.6
Meta	Llama 3.2	1	13.5
		3	34.3

Table A.6: Lightweight LLMs (<4B parameters) benchmarks. Source: [87, 94–96]

Appendix B

Datasets

B.1 Sampled 2000 images from SemArt

In Figure B.1, *Others* (123) aggregates timeframes with fewer than 100 artworks, including 1401–1450 (84), 1301–1350 (66), and 1351–1400 (37), as well as earlier periods with very few artworks.

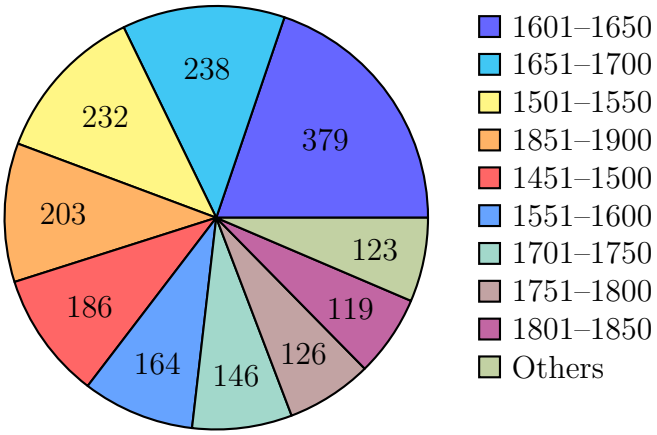

Figure B.1: Distribution of sampled artworks by timeframe.

In Figure B.2, *Others* (279) includes all artistic schools with fewer than 100 artworks, such as English (54), Netherlandish (26), Austrian (15), Hungarian (14), and several others with smaller representation.

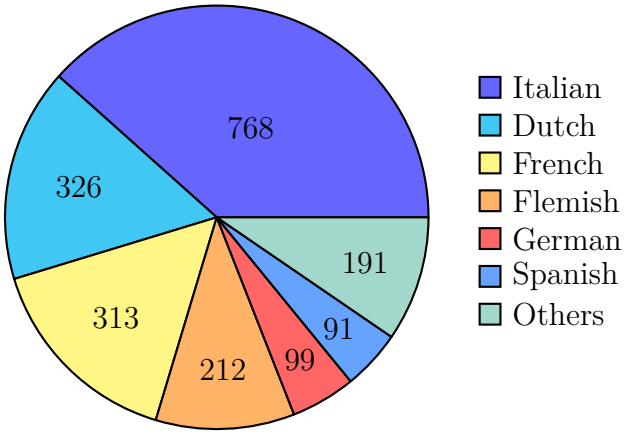

Figure B.2: Distribution of sampled artworks by artistic school/origin.

B.2 Sampled 15,570 images from SemArt

In Figure B.3, *Others* (621) aggregates timeframes with fewer than 400 artworks, including 1351–1400 (378), 1251–1300 (132), 1201–1250 (27), and 1151–1200 (22).

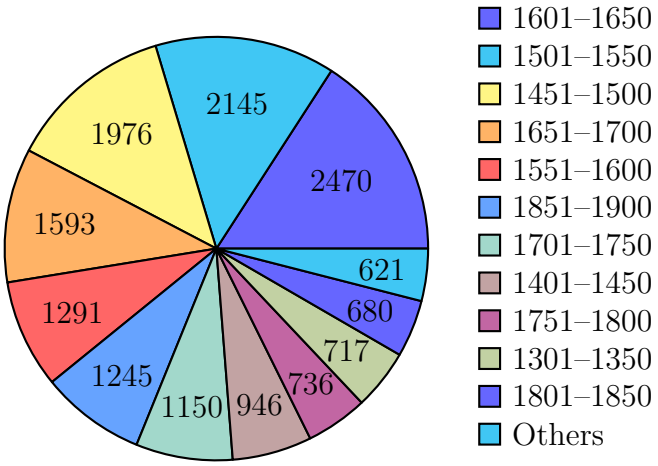

Figure B.3: Distribution of sampled artworks by timeframe.

In Figure B.4, *Others* (1223) includes all artistic schools with fewer than 400 artworks, such as Netherlandish (220) Austrian (168), Hungarian (145), American (81), Danish (75), Swiss (59), Russian (47), and others with smaller representation.

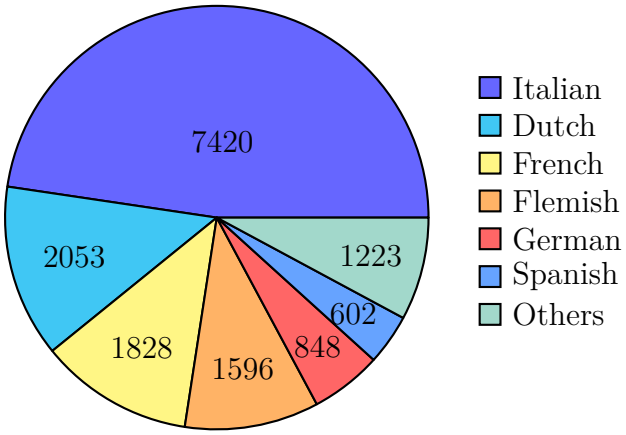

Figure B.4: Distribution of sampled artworks by artistic school/origin.

Appendix C

Similarity scores

C.1 Statistical information between models

This appendix presents a detailed comparison of similarity scores between various text embedding models and state-of-the-art vision-language models.

Embedding	Model	Mean	Std	Min	Max
MiniLM-L6-v2	llava-v1.5-7b	0.2984	0.1394	-0.1046	0.7332
	llava-v1.5-13b	0.3013	0.1367	-0.0957	0.7221
	llava-v1.6-vicuna-7b	0.3609	0.1375	-0.0415	0.7876
	llava-v1.6-vicuna-13b	0.3679	0.1387	-0.0791	0.7561
	llava-v1.6-mistral-7b	0.3613	0.1414	-0.0966	0.8112
	Llama-3.2-11B-Vision-Instruct	0.3737	0.1417	-0.0435	0.7474
	Qwen2.5-VL-3B-Instruct	0.3755	0.1456	-0.0533	0.8427
	Qwen2.5-VL-7B-Instruct	0.3994	0.1435	-0.0672	0.8191
bge-large-en-v1.5	llava-v1.5-7b	0.5529	0.0813	0.3359	0.8191
	llava-v1.5-13b	0.5526	0.0797	0.3253	0.8541
	llava-v1.6-vicuna-7b	0.5997	0.0791	0.3696	0.8496
	llava-v1.6-vicuna-13b	0.6053	0.0788	0.3713	0.8718
	llava-v1.6-mistral-7b	0.5981	0.0787	0.3698	0.8599
	Llama-3.2-11B-Vision-Instruct	0.6084	0.0797	0.3699	0.8447
	Qwen2.5-VL-3B-Instruct	0.6007	0.0803	0.3861	0.8572
	Qwen2.5-VL-7B-Instruct	0.6219	0.0789	0.3839	0.8800
e5-large-v2	llava-v1.5-7b	0.7559	0.0390	0.6640	0.9035
	llava-v1.5-13b	0.7564	0.0377	0.6720	0.8931
	llava-v1.6-vicuna-7b	0.7689	0.0378	0.6682	0.8850
	llava-v1.6-vicuna-13b	0.7697	0.0380	0.6557	0.9100
	llava-v1.6-mistral-7b	0.7667	0.0376	0.6780	0.9116
	Llama-3.2-11B-Vision-Instruct	0.7708	0.0390	0.6497	0.8997
	Qwen2.5-VL-3B-Instruct	0.7700	0.0405	0.6595	0.9161
	Qwen2.5-VL-7B-Instruct	0.7725	0.0407	0.6603	0.9134
gte-large	llava-v1.5-7b	0.8137	0.0370	0.7134	0.9331
	llava-v1.5-13b	0.8145	0.0361	0.7169	0.9437
	llava-v1.6-vicuna-7b	0.8349	0.0356	0.7328	0.9425
	llava-v1.6-vicuna-13b	0.8375	0.0349	0.7284	0.9477
	llava-v1.6-mistral-7b	0.8328	0.0356	0.7277	0.9338
	Llama-3.2-11B-Vision-Instruct	0.8372	0.0363	0.7192	0.9390
	Qwen2.5-VL-3B-Instruct	0.8366	0.0363	0.7376	0.9474
	Qwen2.5-VL-7B-Instruct	0.8434	0.0359	0.7415	0.9637

Table C.1: Updated similarity statistics (mean, std, min, max) between text embeddings and vision-language models.

C.2 Statistical information between pixel max sizes

This appendix presents a detailed comparison of similarity scores between pixel max sizes in the model chosen.

Embedding	Model	Mean	Std	Min	Max
MiniLM-L6-v2	Qwen2.5-VL-7B-Instruct (512×512)	0.3994	0.1435	-0.0672	0.8191
	Qwen2.5-VL-7B-Instruct (512×640)	0.3975	0.1432	-0.0902	0.7890
	Qwen2.5-VL-7B-Instruct (512×896)	0.3978	0.1443	-0.0662	0.7904
	Qwen2.5-VL-7B-Instruct (512×1024)	0.3973	0.1456	-0.0819	0.8300
	Qwen2.5-VL-7B-Instruct (1024×1024)	0.3976	0.1422	-0.0841	0.8397
bge-large-en-v1.5	Qwen2.5-VL-7B-Instruct (512×512)	0.6219	0.0789	0.3839	0.8800
	Qwen2.5-VL-7B-Instruct (512×640)	0.6226	0.0795	0.3807	0.8413
	Qwen2.5-VL-7B-Instruct (512×896)	0.6223	0.0787	0.3736	0.8529
	Qwen2.5-VL-7B-Instruct (512×1024)	0.6221	0.0777	0.3821	0.8488
	Qwen2.5-VL-7B-Instruct (1024×1024)	0.6214	0.0764	0.3792	0.8557
e5-large-v2	Qwen2.5-VL-7B-Instruct (512×512)	0.7725	0.0407	0.6603	0.9134
	Qwen2.5-VL-7B-Instruct (512×640)	0.7734	0.0402	0.6615	0.9199
	Qwen2.5-VL-7B-Instruct (512×896)	0.7732	0.0407	0.6606	0.9233
	Qwen2.5-VL-7B-Instruct (512×1024)	0.7736	0.0402	0.6566	0.9076
	Qwen2.5-VL-7B-Instruct (1024×1024)	0.7729	0.0403	0.6760	0.9118
gte-large	Qwen2.5-VL-7B-Instruct (512×512)	0.8434	0.0359	0.7415	0.9637
	Qwen2.5-VL-7B-Instruct (512×640)	0.8434	0.0362	0.7285	0.9517
	Qwen2.5-VL-7B-Instruct (512×896)	0.8436	0.0362	0.7313	0.9495
	Qwen2.5-VL-7B-Instruct (512×1024)	0.8433	0.0359	0.7432	0.9438
	Qwen2.5-VL-7B-Instruct (1024×1024)	0.8433	0.0355	0.7371	0.9511

Table C.2: Summary of the similarity statistics for Qwen2.5-VL-7B-Instruct evaluated at various image resolutions.

Appendix D

Survey answers

The table below provides supplementary data for the 18 image comparison sets used in the user survey. Negative values indicate a preference for the model using GTE-Large, while positive values indicate a preference for the alternative model, ConvNeXt.

Phrase	Comparison	Mean	Std	Min	Max
Phrase 1	1	-0.52	0.94	-1.66	1.59
	2	-0.17	1.03	-2.27	1.57
	3	-0.44	0.81	-1.75	1.87
Phrase 2	1	-0.13	1.00	-1.66	1.74
	2	-0.34	1.17	-2.10	2.18
	3	0.19	0.83	-1.39	2.02
Phrase 3	1	0.29	1.05	-1.93	2.02
	2	0.09	1.00	-1.80	1.62
	3	-0.13	1.25	-1.93	2.53
Phrase 4	1	0.21	1.03	-1.56	2.19
	2	0.41	0.84	-1.39	1.62
	3	-0.66	1.05	-2.10	1.51
Phrase 5	1	0.13	1.03	-2.02	1.62
	2	-0.51	0.84	-1.67	2.07
	3	-0.16	0.67	-1.28	1.42
Phrase 6	1	-0.67	0.93	-1.93	2.18
	2	-0.06	0.70	-1.39	1.45
	3	0.88	0.60	-0.20	1.92

Table D.1: Summary of participant preferences across phrases and comparisons. Ratings range from 1 (strong preference for left) to 10 (strong preference for right), centered around 0.

Appendix E

FAISS configurations

Table E.1 summarizes the specific FAISS index configurations used in the CPU and GPU benchmarking experiments. Parameters such as the number of clusters (nlist), the number of subquantizers (m_pq), and the number of bits per subquantizer (nbits_pq) were selected based on dataset size and hardware constraints to ensure training and fair comparison.

# Samples	Index Type	nlist	m_pq	nbits_pq (CPU)	nbits_pq (GPU)
100	IndexIVF	4	–	–	–
100	IndexIVFPQ	4	4	4	–
1000	IndexIVF	25	–	–	–
1000	IndexIVFPQ	25	8	6	8
5000	IndexIVF	50	–	–	–
5000	IndexIVFPQ	50	16	6	8
10000	IndexIVF	100	–	–	–
10000	IndexIVFPQ	100	16	6	8
15000	IndexIVF	100	–	–	–
15000	IndexIVFPQ	100	16	6	8

Table E.1: Index configurations used for IndexIVF and IndexIVFPQ in CPU benchmarking.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**