Politecnico di Torino

Politecnico di Torino

Computer Engineering

A.a. 2024/2025

Graduation Session October 2025

Integrating Multi-Modal Reasoning and Explainable AI for Dermatological Image Analysis via LLM-Orchestrated Toolchains

Supervisors:
Flavio Giobergia
Ignazio Gallo

Candidate:
Leonardo Sgroi

Table of Contents

Abstract 1

Introduction 2

1 Background and Related Works 5
- 1.1 Large Language Models - LLMs . . . 5
  - 1.1.1 Transformer Architecture . . . 6
- 1.2 Agentic AI and AI Agents . . . 8
- 1.3 Explainable AI . . . 10
  - 1.3.1 Gradient-Based Methods . . . 12
  - 1.3.2 Perturbation-based Methods . . . 14
  - 1.3.3 Concept-based Methods . . . 17
- 1.4 Detection Models . . . 20
- 1.5 AI in Dermatology . . . 21

2 Methodologies 23
- 2.1 Agent Structure . . . 23
- 2.2 Classification Tool . . . 28
  - 2.2.1 Classification Model . . . 30
  - 2.2.2 Datasets . . . 30
  - 2.2.3 Concept Bottleneck Model . . . 31
- 2.3 Embedding Tool . . . 32
- 2.4 Saliency Tool . . . 34
- 2.5 Detection Tool . . . 37

3 Experiments 40
- 3.1 Reasoning Core – Large Language Model . . . 41
  - 3.1.1 Experiment Setup . . . 41
  - 3.1.2 Results . . . 43
- 3.2 Embedding Tool - Text Embeddings for Metadata . . . 45
  - 3.2.1 Datasets . . . 45

3.2.2 Experiment Setup 45
3.2.3 Results 47
3.3 Classification Tool - CBM 48
3.3.1 Training Strategies 48
3.3.2 Dataset 48
3.3.3 Metrics 49
3.3.4 Loss Functions 49
3.3.5 Results 50
3.3.6 Discussion 51
3.4 Saliency Tool - Explainable AI Techniques 52
3.4.1 Experiment Setup 52
3.4.2 Results 55
3.4.3 Error Analysis and Central Agent Perspective 55
3.4.4 Evaluation of SmoothGrad 56

4 Conclusions 57

Bibliography 59

Abstract

Skin cancer is one of the most common and dangerous kinds of cancer worldwide, although its detection remains a challenge even for expert dermatologists. This thesis explores how artificial intelligence can be a trusted assistant in the diagnostic process by combining the reasoning power of Large Language Models (LLMs) with the precision of state-of-the-art vision tools.
The proposed framework is a modular agent with a central reasoning core that leverages a set of specialized tools for image classification, lesion detection, patient metadata integration, and explainable AI. Through extensive experiments, the thesis evaluates the contribution of each component. First of all, the ability of multimodal language models, such as GPT-4o and Gemini, is analyzed both in classifying dermatological images with their own vision and in interacting with vision tools. Furthermore, the research focuses on the integration of patient information via text embeddings into the classification model, to understand whether this data can enhance the performance of the tool.
The project also includes an evaluation on the interpretability gains given by concept-based explainability methods; by exploiting the annotations of the SkinCon dataset, made of 48 clinical concepts annotated by dermatologists, the model learns to predict these concepts before performing the final classification. This technique helps the central LLM to better communicate with the user by supporting its answers with a set of "proofs." The results show that while general-purpose AI struggles on its own in fine-grained medical tasks, combining it with domain-specific tools significantly boosts performance and reliability. This AI agent interacts, through a ReAct loop approach, with dermatologists in natural language, providing meaningful explanations of its decisions in order to make the reasoning transparent to the user. This work demonstrates the potential of combining LLM reasoning with modular vision tools to build an effective dermatological AI assistant. The agent mimics the behaviour of a clinician by delegating tasks to specialized tools and integrating their outputs into a final decision. Its modularity allows for easy expansion and integration into real-world clinical workflows. The code of the agent and the experiments is available at the following GitHub repository: https://github.com/Sgroi71/MasterThesis-DermAgent.

Introduction

Over the past few years, the integration of Artificial Intelligence (AI) in the medical domain has shown promising results in the analysis of clinical images, especially in visually demanding fields like dermatology. Skin cancer and related diseases represent some of the most common health issues worldwide. In this context, the ability to provide an early and accurate diagnosis is of fundamental importance, since detecting a lesion at an initial stage can greatly improve treatment outcomes and even save lives.
Despite the remarkable success of the vision models for medical image and the availability of large datasets, the process of diagnosing skin lesion remains complex. Moreover, the diagnostic process is not free from subjective interpretation, and even among experts of this field discrepancies emerge. This is where AI technologies can make tangible differences.
Large Language Models (LLMs) have shown remarkable capabilities in processing natural language and reasoning across different type of information. When LLMs are combined with specialized vision-based tools, they cannot only analyse the image giving a classification into a fixed set of pathologies but also explain their reasoning, thus becoming a reliable assistant for experts.
The challenge is to design systems that are not only accurate but also transparent, modular and adaptable to the workflow of clinicians practice. In other words, the application of AI in medicine cannot be "black box": it needs to provide explanations, justifications, and interaction modes that integrate clinicians in the decision-making of the system, making sure that it is coherent with the way doctors work.

The proposed thesis has the ambition to create an AI agent capable of assisting dermatologists in their daily activities. The resulting AI assistant will not replace the human expert but will support them in repetitive actions, suggesting interpretable decisions, and enabling real-time interactions with the AI assistant. This aligns with the current shift in AI research toward collaborative and human-in-the-loop systems, making this work not only technically advanced but also clinically impactful.

Goal

The primary goal of this thesis is to design and implement an AI agent for dermatological diagnosis that leverages both multimodal reasoning and vision-based tools, while experimenting different Explainable AI techniques to the latter. Specifically the objectives are:

- Developing a tool that integrates the patient information (metadata) to the features of the image extracted from the vision model;
- Integrating a specialized tool for lesion detection and classification, ensuring a transparent decision-making process;
- Evaluating the effectiveness of saliency-based explanation methods and the concept bottleneck approach [1] in improving interpretability for clinicians;
- Implementing the AI agent and benchmark the system across different LLMs to assess their ability to reason and interact within the medical context;
- Development of a user-friendly interface, enabling flexible deployment options from local to cloud-based solutions.

Thesis Structure

To achieve these goals, the thesis is organized into five main chapters, each addressing a key aspect of the research path.

- **Chapter 1 (Background and Related Works)**: This chapter combines a review of the relevant literature with the presentation of the theoretical foundations that underpin this thesis. It begins with an overview of Large Language Models (LLMs), describing their architecture, history, and limitations in medical tasks. Subsequently, it discusses the notion of AI agents and the broader paradigm of Agentic AI, with particular attention to their adoption in the clinical context and their modular structure. The chapter also introduces the main approaches in Explainable AI (XAI), focusing on methods that allow machine learning models to produce interpretable explanations and the strategies used by researchers to evaluate them. Finally, it presents detection models, which play a central role in identifying dermatological lesions within images, and reviews how AI has been applied in dermatology, especially in the analysis of skin lesions and cancer detection. Together, these concepts provide the necessary background to understand the originality of the proposed approach and frame the methodologies and experiments described in the following chapters.

- **Chapter 2 (Methodologies)**: This chapter describes in detail the design and implementation of the AI agent and its modular architecture: detection, classification and explanation tools, as well as the integration of the concept bottleneck model.
- **Chapter 3 (Experiments)**: Here it is presented the experiment setup, including datasets, evaluation metrics and the discussion about the results.
- **Chapter 4 (Conclusions)**: The final chapter summarizes the main findings, underlining the contribution of this works, its limitations and potential directions for future research.

Chapter 1

Background and Related Works

In this chapter, we present an overview of both the relevant literature and the theoretical foundations that underpin this research. The discussion combines an examination of prior studies with the explanation of the fundamental concepts required to contextualize the proposed framework. To provide clarity and structure, the chapter is organized into five subsections: 1.1 Large Language Models, exploring their history and structure; 1.2 Agentic AI and AI agents, focusing on their architecture, applications, and relevance in healthcare; 1.3 Explainable AI (XAI), underlining its contribution to transparency, trust, and clinical adoption; 1.4 Detection models, detailing approaches for lesion localization and analysis in dermatological imaging; and finally, 1.5 AI in dermatology, with emphasis on skin lesion analysis and cancer detection.

1.1 Large Language Models - LLMs

The development of Large Language Models (LLMs) represents one of the most significant advancements in artificial intelligence over the past decade. Before discussing Large Language Models, we need to introduce the broader history of language modelling. Traditional language models were based on probabilistic methods; they are also called n-gram models and they estimate the probability of a word to be chosen, given the preceding ones.
While effective in small contexts, n-gram models have some limitations:

- As n increases, the number of possible n-grams grows exponentially;
- Lack of ability to capture long-term dependencies in text;

- Lack of semantics, so similar words are treated in the same way as completely different ones.

Moving beyond n-gram models, researchers started in 2000 to use neural networks to learn language models [2]. This shift was marked by the development of word embeddings (Word2Vec [3]) and sequence-to-sequence (seq2seq) models using LSTM. The seq2seq architecture was introduced to handle tasks with variable-length input and output sequences, such as machine translation. It is composed of two main parts:

- **Encoder**: A recurrent network (typically LSTM or GRU) that compresses the input sequence into a context vector.
- **Decoder**: Another recurrent network that, starting from the context vector, generates the output sequence step by step.

Despite its early success, the model faced two major limitations: the bottleneck problem, i.e., the difficulty of representing long sequences with a single fixed vector, and the lack of parallelization, since RNNs process tokens strictly one after another.

1.1.1 Transformer Architecture

A decisive turning point came in 2017 with the introduction of the Transformer architecture by Vaswani et al. [4] in "Attention is All You Need." Unlike RNNs, Transformers rely entirely on a self-attention mechanism, allowing models to capture dependencies between tokens regardless of their distance in a sequence. This innovation solved the "bottleneck" problem and enabled efficient parallelization and scalability.
As shown in figure 1.1, the Transformer architecture is typically composed of:

- **Embedding Layer**: Converts tokens into dense vector representations.
- **Positional Encoding**: Adds sequence order information, to solve the attention mechanism's lack of inherent sequentiality.
- **Stacked Transformer blocks**, where each block includes:
  - **Multi-head self-attention** to capture diverse relational patterns among tokens.
  - **Feed-forward neural networks** applied independently to each position.
  - **Residual connections and layer normalization** for stability and gradient flow.

- **Output Layer**: Produces probability distributions over the vocabulary, enabling autoregressive text generation.

There are three Transformer variants:

- **Encoder-Only Transformers** (Figure 1.2): Composed solely of stacked encoder layers, each with a self-attention layer. This structure is ideal for tasks like classification and masked language modelling. An example of this category is BERT [5], which is trained to predict masked words within the context of a sentence.
- **Encoder-Decoder Transformers**: Combine an encoder architecture, that handles the input, with a decoder architecture that generates the output. It includes the self-attention layer in the encoder and the cross-attention layers where the decoder attends to encoder outputs. An example is T5 (Text-to-Text Transfer Transformer) [6], great for tasks that translate or transform one sequence into another. The drawback is the heavier computation due to the dual-stack architecture.
- **Decoder-Only Transformers**: Consist exclusively of decoder layers—each containing causally masked self-attention and feed-forward sublayers—without any encoder component. They are ideal for generative tasks like chatbots or creative writing.

On top of the latter architecture, OpenAI introduced Generative Pre-trained Transformers (GPT), inaugurating what is now called the GPT era.
GPT-1 (2018) demonstrated the effectiveness of large-scale pre-training followed by task-specific fine-tuning. GPT-2 (2019), based on the previous version but with a larger training corpus (10x) and a larger model (10x), removed the fine-tuning phase, showing emerging capabilities to solve problems it was not explicitly trained on. GPT-3 (2020), with 175 billion parameters, exhibited few-shot and zero-shot learning capabilities. Subsequent models, including GPT-4 and multimodal variants like GPT-4o, expanded these capabilities by integrating text, images, and other modalities.

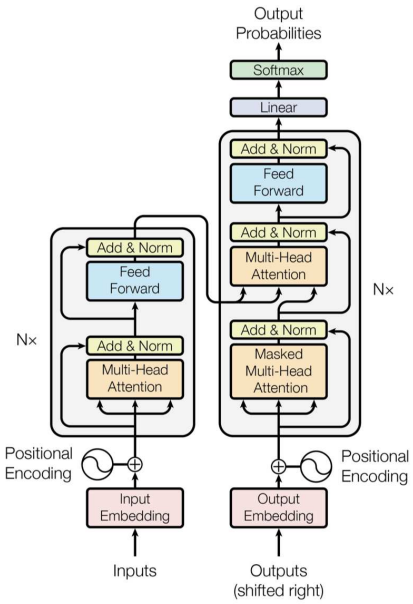


Figure 1.1: Transformer architecture [4]

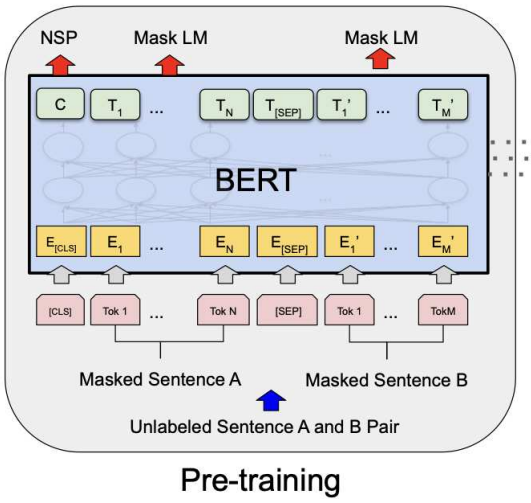


Figure 1.2: BERT architecture [5]

1.2 Agentic AI and AI Agents

Artificial Intelligent Agents (AI Agents) and the broader paradigm of Agentic AI have recently gained increasing attention as frameworks for building autonomous systems. Although the two terms are often used interchangeably, they describe distinct design philosophies and capabilities.

As highlighted in the conceptual taxonomy of Sapkota et al. [7], "AI Agents are autonomous software programs that perform specific tasks", usually by combining perception, reasoning, and tool-use. For instance, MedRAX, introduced by Fallahpour et al. [8], is a domain-specific AI Agent tailored for chest X-ray interpretation, capable of orchestrating specialized vision models through a reasoning loop.

In contrast, Agentic AI represents a paradigm shift. Rather than focusing on isolated agents, Agentic AI coordinates multiple specialized agents capable of decomposing complex objectives into smaller tasks and dynamically collaborating within a workflow. Acharya et al. [9] define it as "autonomous systems designed to pursue complex goals with minimal human intervention", showing adaptability and long-term planning capabilities in evolving environments.

Following this paradigm, Feng et al. [10] proposed M3Builder, a multi-agent system that automates medical imaging workflows by coordinating four role-specific agents (task manager, data engineer, module architect, and model trainer), achieving a

success rate above 90% in benchmark tasks. Similarly, Shahin et al. [11] argue that agentic workflows combined with specialized AI Agents can accelerate drug development pipelines and quantitative clinical pharmacology.
The concept of an AI agent also emerges from the attempt to extend Large Language Models (LLMs) beyond static workflows into systems capable of autonomous reasoning and decision-making.
Agentic behaviour introduces the ability for the model to decide its own control flow to solve complex problems. This flexibility enables the system to dynamically adapt to tasks, making the LLM not just a reactive tool, but a proactive agent.
Before full agentic systems, LLMs were typically organized into structured chains, such as sequential, tree, or router chains. Agentic systems, instead, can set their own goals, plan actions, and adapt to changing circumstances, thus displaying initiative and continuous improvement.
A full agent architecture (Figure 1.3) generally includes several components:

- **Agent (Core Controller)**: Central decision-maker orchestrating the process.
- **Tools**: External modules (e.g., calculator, code interpreter, search engine) used to extend capabilities.
- **Memory**: Divided into short-term (temporary context) and long-term (persistent user or task knowledge).
- **Planning**: Strategies to break down complex objectives into subgoals.
- **Reflection and Self-Criticism**: Mechanisms for evaluating past decisions and improving future reasoning.
- **Chain-of-Thought & Subgoal Decomposition**: Stepwise reasoning and structured task division.

When multiple agents are deployed, interaction schemes define their behaviour:

- **Cooperative**: Agents collaborate, sharing information to achieve a shared goal.
- **Adversarial**: Agents compete, optimizing strategies against opponents.
- **Mixed**: Combinations of cooperation and competition, organized in parallel or hierarchical structures.

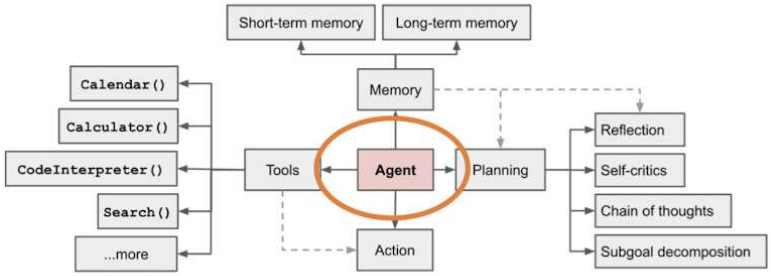


Figure 1.3: General architecture of an Agentic AI system.

1.3 Explainable AI

Cino et al. pointed out physicians' skepticism about the use of AI in the medical field, due to the "black box" nature of AI models. They emphasized that "the lack of transparency and explainability inherent in these models hinders their widespread acceptance in clinical settings" [12].
Recent studies have integrated explainable AI techniques such as Grad-CAM [13] and t-SNE [14] to provide visual justification for model predictions [12]. Moreover, novel frameworks like SkinGen [15] explore interactive vision-language models to generate synthetic but realistic visual interpretations of possible diagnoses, aiming to enhance user trust and comprehension.

Explainable AI (XAI) provides methods to make models more interpretable, allowing humans to understand the reasoning behind predictions. Explainability is not a single step but spans the entire machine learning pipeline:

- **Pre-modelling explainability**: Conducted before training, it focuses on understanding and preparing data. Typical activities include exploratory data analysis, interpretable feature engineering, and dataset documentation.
- **In-modelling explainability**: Here, models are designed to be inherently interpretable. Examples include decision trees, linear models, or concept-based approaches such as concept bottleneck models.
- **Post-hoc explainability**: Applied after model training, it provides explanations for black-box models. Common methods include gradient-based approaches, surrogate models, and counterfactuals.

XAI methods can differ in their scope:

- **Global explanations**: Aim to describe the overall behaviour of a model. Examples include permutation feature importance, partial dependence plots, or decision trees (if small enough to be interpretable).
- **Local explanations**: Focus on individual predictions, answering the question "why did the model decide this class for this instance?".

In this thesis we will focus only on methods of the latter category, since the model to explain is a vision-based model applied to medical images. In this context, looking for general rules that explain the overall behaviour of the model is not trivial.
Another important distinction is based on generalizability:

- **Model-agnostic methods** (e.g., LIME, SHAP) treat the model as a black box and can be applied to any architecture, since they only need access to the inputs and the outputs.
- **Model-dependent methods** exploit the internal structure of a specific model, such as Grad-CAM, which can be applied only to CNNs. These often provide deeper insights but are limited in scope.

One of the major lines of research focuses on feature attribution methods. An example of this category is SHAP [16], derived from cooperative games theory, which assigns contributions to each input feature and has become a reference point for model-agnostic explainable techniques.
Covert et al. [17] proposed a unified framework of removal-based explanations, showing that in many contexts, predictions can be explained through the principle of simulating feature removal.
Another family of methods explores prototype and concept-based explanations. XProtoNet [18], for example, learns disease-representative prototypes to provide both global and local explanations. Similarly, Concept-Bottleneck Models (CBMs) impose predictions through human-defined intermediate concepts, enabling direct interventions by the expert.
Koh et al. [1] state: "By construction, we can intervene on these concept bottleneck models by editing their predicted concept values and propagating these changes to the final prediction". This paradigm shifts model interaction closer to clinical reasoning, where physicians rely on explicit intermediate concepts.
Many of the explainable methods representation applied to vision-based models uses saliency maps: for example, Ancona et al. [19] compare many attributions methods based on gradient-information with respect to the input.
Recent works have also presented techniques to evaluate the interpretability of

saliency maps: Bokadia et al. [20] proposed perceptual and semantic interpretability metrics, showing that faithfulness alone is insufficient. Their study on melanoma concluded that "None [of the saliency methods] achieves high scores across all three metrics [...] but different methods perform well in different regions of the data distribution".

1.3.1 Gradient-Based Methods

The most widely adopted post-hoc model-agnostic methods are gradient-based techniques, which leverage the derivative of the model's output with respect to its input features. The intuition is that the gradient indicates how small changes in the input affect the model's predictions.
Generally, the generated explanation has the same size as the input and, in the case of images, it is shown as a saliency map, highlighting the pixels most influential for the decision.
Gradient-based methods differ in how the gradient is computed:

- **Vanilla Gradient**: Computes the gradient of the output score for a target class with respect to the input features. Formally, given a model F trained for C classes, the output of the model for the input I is $F(I) = [F_1(I), \dots, F_C(I)]$. The objective, given an input I_0 with p features, is to compute a relevance score for each feature and for class c: $R^c = [R^c_1, \dots, R^c_p]$.
  The idea is to model the score function F_c as a linear function, but since it is non-linear we approximate it with the first-order Taylor expansion:

  $$F_c(I) \approx w^T \cdot I + b = R_c^T \cdot I + b$$

  Where R_c is the derivative of the score and it is calculated as follows:

  $$\nabla_x F_c(x) = \left[\frac{\partial F_c}{\partial x_1}, \dots, \frac{\partial F_c}{\partial x_p}\right]$$

  As done during the training phase of the model, these derivatives are calculated via backpropagation. While simple and computationally efficient, Vanilla Gradients are often noisy and unstable, as small perturbations in the input can cause large variations in the gradient.

- **Input $\times$ Gradient**: An extension of the basic approach, where the gradient is multiplied element-wise with the input itself. This highlights features that not only have a strong gradient but also high activation in the input. In practice, it often produces sharper and more interpretable attribution maps compared to Vanilla Gradients.

- **Integrated Gradients (IG)**: Compares the model's prediction on an input x with a baseline input x' (often a zero vector). The attribution is computed by integrating the gradients along the straight-line path between x' and x, as shown in Figure 1.4. Formally:

$$IG_i(x) = (x_i - x_i') \times \int_0^1 \frac{\partial F\Big(x' + \alpha(x - x')\Big)}{\partial x_i} \, d\alpha$$

Where α is the interpolation constant.
Let k be a scaled feature perturbation constant and m the number of steps; we can approximate the integral as follows:

$$IG_i^{(m)}(x) = (x_i - x_i') \, \frac{1}{m} \sum_{k=1}^{m} \frac{\partial F\Big(x' + \frac{k}{m}(x - x')\Big)}{\partial x_i} \qquad [21]$$

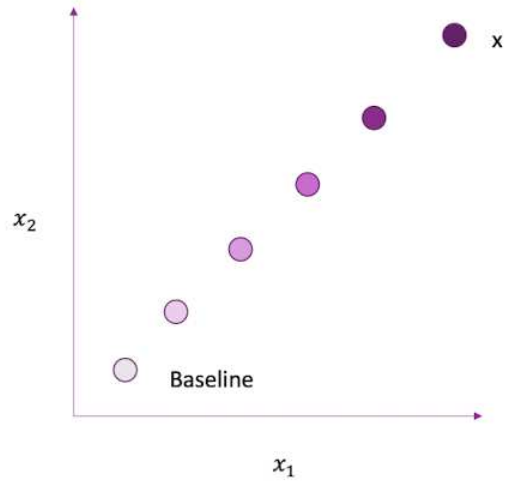
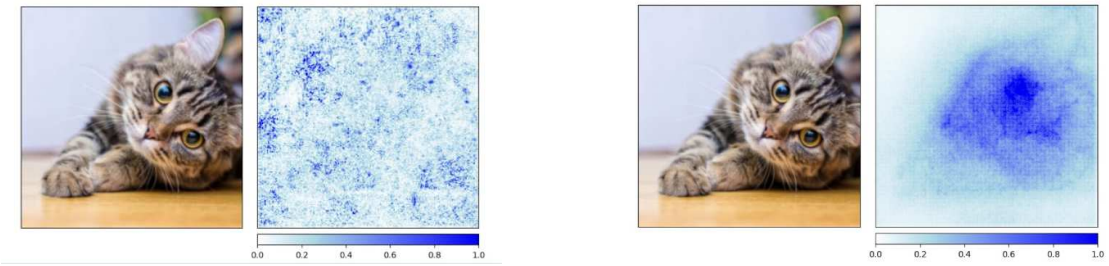


Figure 1.4: Integrated gradient baseline

One of the major limitation of the above methods is that they are noisy, because slight variations in the input data can result in significant changes in the model output (and thus in the gradients), leading to noisy gradients and instability.
SmoothGrad reduces noise by generating N versions of the same input by adding Gaussian noise and then averaging them:

$$SG(x) = \frac{1}{N} \sum_{i=1}^{N} \nabla_x F(x + \epsilon_i), \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

This process smooths out fluctuations and results in clearer and more stable saliency maps. In the figure 1.5 we can see the comparison between the vanilla gradient techniques with and without Smooth-Grad.



Vanilla Gradient

Vanilla Gradient + Smooth-Grad

Figure 1.5: Comparison between Vanilla Gradient with and without Smooth-Grad

1.3.2 Perturbation-based Methods

Another important family of post-hoc explainability approaches are removal-based or perturbation-based techniques, also known as occlusion methods. Their central idea is to evaluate the impact of removing or altering parts of the input on the model's prediction.
By quantifying how predictions change when features are hidden, it is possible to infer their relative importance.
The general functioning of perturbation-based methods is formalized as follows. Given a model f and an input x, the importance value of the feature A_i is estimated by comparing the prediction with and without that feature:

$$\text{Importance}(A_i) = f(x) - f(x \setminus A_i)$$

where $x \setminus A_i$ denotes the input with feature A_i removed or perturbed. Different strategies exist for simulating removal:

- Zeroing (set removed features to zero).
- Default values (replace features with mean or constant values).
- Blurring (for image regions).
- Marginalization over the feature distribution.

SHAP (SHapley Additive exPlanations) builds on the concept of Shapley values from cooperative game theory. In this analogy:

- Each **feature** is a "player";
- The **prediction** is the "payout" of the coalition of features;
- The Shapley value ϕ_i measures the average marginal contribution of feature i across all possible feature coalitions.

Formally, for a feature i in the coalition S:

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \Big(v(S \cup \{i\}) - v(S) \Big)$$

where $v(S)$ is the total expected sum of payoffs the players in S can obtain by cooperating. It is calculated as $v(S) = f(S) - E[f(X)]$, where $f(S)$ is the model prediction marginalizing over the features not in S.

This technique has some limitations in terms of computation time. In fact, it is exponential in the number of feature values since we need to evaluate the instance over all possible coalitions.
The proposed solution is the approximation with Monte-Carlo sampling: take a random instance z from the dataset, and define x_{+j} as the instance x where a random number of feature values are replaced by feature values from the random data point z; the value for the feature j is kept as in the original feature x. Conversely, x_{-j} is defined as x_{+j} but with the value of j also replaced by the value of the sampled z.

At each iteration the marginal contribution is computed as $\phi_j^m = f(x_{+j}) - f(x_{-j})$. Finally, after M iterations we can compute the average of the marginal contributions.

SHAP proposes two approaches for estimating the Shapley values:

- **KernelSHAP**: Approximation using weighted linear regression that is model-agnostic.
- **TreeSHAP**: Efficient exact computation for tree models.

The output of SHAP can be feature attributions for each instance if we want local explanations, or it can be aggregated into global insights as shown in Figure 1.6.

In the case of image data, the image is divided into many parts called superpixels. We define the coalition of the features with an interpretable representation: a vector where each element is a binary value indicating if the superpixel is taken or not. Figure 1.7 shows this method graphically. To reduce the dimensionality of the

input space and produce explanations at a semantically meaningful level, images were segmented into superpixels using the **SLIC (Simple Linear Iterative Clustering)** algorithm [22].

SLIC is inspired by the k-means clustering method. The algorithm does not only consider color similarity but also the spatial distance between pixels. This way, superpixels are compact and follow the contours of visual structures. The distance metric used is:

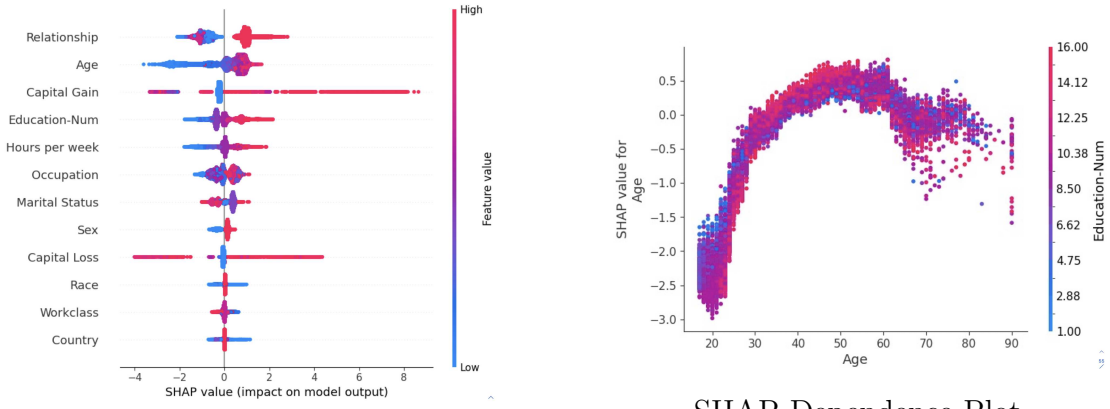
$$D = \sqrt{d_c^2 + \left(\frac{d_s}{S}\right)^2 \cdot m^2}$$

Where:

- d_c: color distance
- d_s: spatial distance
- S: average superpixel size
- m: compactness parameter (high $\rightarrow$ more regular superpixels; low $\rightarrow$ more adapted to image edges)

The image is initially divided into a regular grid, and in each block a center (seed) is chosen. The number of blocks corresponds to the desired number of superpixels ($n\_segments$). At each iteration, pixels are assigned to the closest centroid considering the combined distance, and the cluster centroids are updated. This is repeated until convergence.

The resulting segmentation mask was converted into a tensor and used as `feature_mask` in SHAP computations, ensuring that attributions were calculated over homogeneous regions rather than individual pixels. A neutral grey baseline was adopted to represent the absence of informative features.



SHAP Summary Plot

SHAP Dependence Plot

Figure 1.6: SHAP global insights: Summary plot and Dependence plot

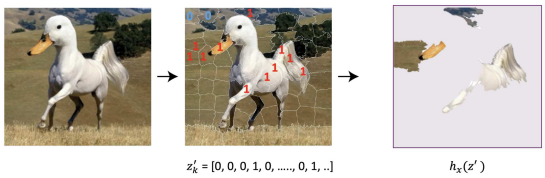


Figure 1.7: SHAP's Interpretable Representation

1.3.3 Concept-based Methods

Traditional XAI methods explored so far provide explanations in terms of features or pixels. However, these representations are often misaligned with the way humans reason, making the explanations difficult to interpret.
Concept-based Explainable AI (C-XAI) aims to bridge this gap by introducing human-understandable concepts into the explanation process.
A concept can be any abstraction, such as a color, a texture, an object, or even a higher-level attribute (e.g., "round," "spotted," or "beak").
Several strategies exist in the literature to define concepts in XAI:

- **Symbolic concepts**: Human-defined attributes (e.g., "the beak of a bird").
- **Unsupervised concepts**: Clusters of similar samples in the latent space.
- **Prototypes**: Representative examples or parts of samples.
- **Textual concepts**: Embeddings derived from textual descriptions of classes.

As shown in Figure 1.9, these concepts can be linked to model behavior through different forms of association:

- **Class-Concept Relations**: Describe the relationship between a specific concept and an output class of the model. They can express concept importance or logic rules involving multiple concepts and their connection to an output class. This type of relation can be applied to all kinds of concepts;
- **Node-Concept Association**: Assigns a concept to an internal unit (or a filter) of a network. It enhances transparency by showing what internal units "see" in a given sample. The association can be defined post-hoc (by analyzing hidden units maximally activating on concept samples) or enforced during training (requiring a unit to predict a concept);
- **Concept Visualization**: Highlights the input features that best represent a specific concept. Similar to saliency maps but at the level of concepts, this is particularly useful when non-symbolic or unsupervised concepts are employed. It helps to understand which attributes or prototypes the network has actually learned.

One of the most widely used methods in C-XAI is the Concept Bottleneck Model (CBM). As we can see from Figure 1.8, extracted from the original CBM paper [1], instead of mapping input x directly to output y, the CBM enforces an intermediate concept layer c that explicitly predicts human-defined concepts before performing the final task prediction:

$$x \longrightarrow g(x) = c \longrightarrow f(c) = y$$

Here, $g(x)$ is the concept encoder, mapping inputs to a set of concept predictions, and $f(c)$ is the task predictor, which takes the predicted concepts and outputs the final class.
Koh et al. [1] proposed different training approaches:

- **Independent**: Train g to predict concepts and f to predict classes separately;
- **Sequential**: Train g first, then use its outputs to train f;
- **Joint**: Train g and f together, balancing task loss and concept loss.

This method has several advantages. In fact, it produces a transparent, by-design model that provides explanations in terms of high-level concepts ("I predict melanoma because the lesion is irregular, dark, and asymmetric"). Moreover, users can edit predicted concepts and observe how predictions change. In this way, expert knowledge can also become part of the AI decision process, increasing the trustworthiness of clinicians.

However, this strategy also has limitations. Enforcing concept prediction may reduce classification accuracy compared to black-box models. Furthermore, CBMs require concept annotations, which are costly to obtain.

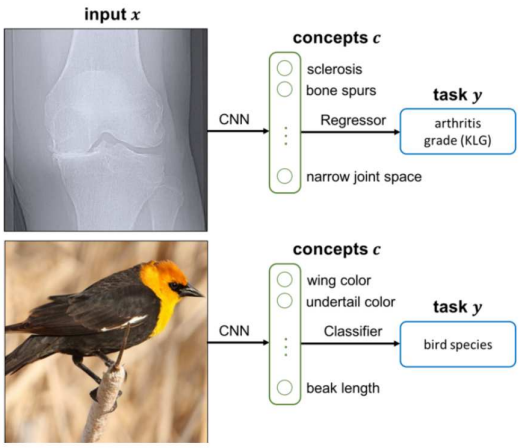


Figure 1.8: Concept Bottleneck Structure [1]

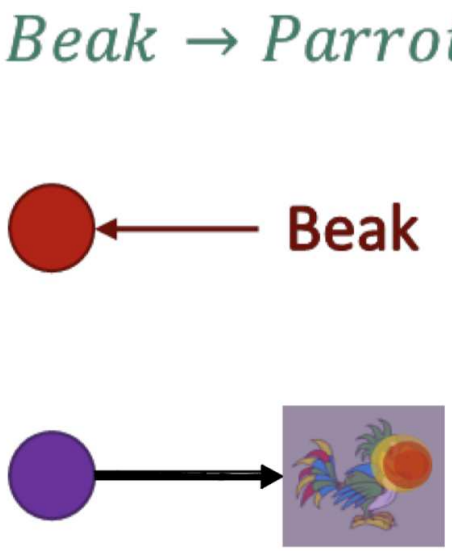


Figure 1.9: Concept Associations

1.4 Detection Models

Object detection is a core task in computer vision. The objective is not only to classify the objects present in an image but also to localize them with bounding boxes. Unlike classification, which assigns a single label to an entire image, detection produces a variable number of predictions, each one with a dual output:

- **Bounding boxes**: Four real numbers, where the first two indicate the position of the upper-left corner, the third represents the width, and the last represents the height.
- **Category label**: A single number that indicates the predicted class within a fixed, known set of categories.

This dual requirement makes detection more challenging and computationally demanding than simple classification. In clinical applications, such as dermatology, it enables the identification of specific regions of interest (e.g., skin lesions) within body images, thus reducing the focus area for subsequent fine-grained analysis.
With the advent of deep convolutional networks, remarkable results were obtained in this field.
In 2014, Girshick et al. [23] presented R-CNN, introducing region proposals combined with CNN-based classification. In the following years, Girshick proposed Fast R-CNN (2015) [24] and Faster R-CNN (2016) [25], models that improved efficiency by integrating proposal generation directly into the network.
Mask R-CNN (2017) [26] extended Faster R-CNN by adding a segmentation branch, becoming the first widely adopted unified model for detection and instance segmentation. While convolution-based methods proved extremely effective, they relied on complex pipelines, anchor design, and non-maximum suppression (NMS). These handcrafted components limited scalability and generalization.
Inspired by the success of Transformers in natural language processing, DETR (2020) redefined object detection as a set prediction problem. Instead of using anchors and post-processing, DETR applied encoder–decoder attention to produce object queries directly matched to ground-truth boxes via bipartite matching.
Although DETR was conceptually elegant, it suffered from slow convergence and suboptimal performance compared to CNN-based models. This led to several improvements:

- **DAB-DETR** [27]: Introduced dynamic anchor boxes.
- **DN-DETR** [28]: Leveraged denoising training to accelerate convergence.
- **DINO (2022)** [29]: Combined these ideas with improved query selection and anchor refinement, achieving state-of-the-art detection on COCO [30].

Despite these advances, a gap remained: the best detection and segmentation models were still separate. DINO excelled at detection, while Mask2Former [31] excelled at segmentation.
To address this, Mask DINO (2023) [32] was proposed as a unified Transformer-based model for both detection and segmentation. Its contributions include:

1. **Mask prediction branch**: Added in parallel to the box prediction head, enabling instance, semantic, and panoptic segmentation.

2. **Unified query selection**: Leverages encoder tokens to generate both masks and boxes, allowing early mask predictions to guide better box initialization.

3. **Unified denoising**: Extends denoising training to masks, improving stability and convergence.

4. **Hybrid matching**: Incorporates classification, box, and mask losses into bipartite matching, ensuring consistency across predictions.

The key contribution of Mask DINO is the demonstration that detection and segmentation can reinforce each other: masks provide pixel-level priors for detection, while detection pretraining on large datasets boosts segmentation performance.
In benchmarks, Mask DINO achieves state-of-the-art results across all segmentation tasks (instance, semantic, and panoptic) while also surpassing DINO in detection.

1.5 AI in Dermatology

Dermatology has become a preferential field for AI due to the reliance on visual data such as dermoscopic and clinical images.
According to Liopyris et al. [33] "CNN algorithms can classify skin lesions from dermoscopic images with superior or at least equivalent performance compared to clinicians". The availability of large public dataset such as the ISIC Archive [34] has significantly accelerated research. Despite these promising outcomes, challenges remain regarding generalizability, image quality, and integration into routine practice.
One of the major concerns is the lack of diversity in training data. Danashjou et al. [35] highlight this gap by releasing the Diverse Dermatology images (DDI) dataset, the first to include expert curated and pathologically confirmed images with diverse skin tone. They show that state-of-the-art algorithms such as DeepDerm [36] and ModelDerm [37] "exhibit substantial limitations on the DDI dataset, particularly on dark skin tones and uncommon diseases.". Specifically, they found that training on diverse data closed the performance gap, demonstrating the critical role of representative datasets in addressing algorithm bias.

Transformer-based models offer better capture of long-range dependencies in images: Gallazzi et al. [38] showed that by merging multiple datasets into a larger one, transformer architectures achieved 86.37% accuracy on standardized test data; this demonstrates that "the Transformer-based architecture achieves state-of the-art performance in skin lesion classification, outperforming traditional CNNs and other DL models previously employed for similar tasks."

Chapter 2

Methodologies

In this chapter, the methodological and technical implementation of the proposed AI agent is described. The focus is on presenting the modular structure of the agent and the tools integrated to achieve multimodal reasoning and explainable decision-making.
The chapter is organized into six main sections. First, the structure of the system is introduced, together with the technical details such as the libraries used for the implementation of the agent, the reasoning core, and the memory component.
Subsequently, the embedding tool is presented, which manages the retrieval of patient metadata and embeds them in a form that can be integrated into the input of the classification model. The classification tool is then analyzed in depth, covering the model architecture, datasets, and the adoption of a concept bottleneck model to enhance explainability. The saliency tool section focuses on the techniques used to generate visual explanations. Finally, the detection stage that localizes multiple lesions at risk within complex dermatological images.
Together, these modules form a unified framework capable of supporting clinicians in their diagnostic workflow.

2.1 Agent Structure

The AI agent is designed as a modular framework that integrates a Large Language Model (LLM), which orchestrates reasoning and tool selection within a ReAct loop. This allows the agent to iteratively decompose a medical query into sequential steps of **observation**, **reasoning**, and **action**, dynamically deciding which tools to employ.

The agent is built on top of the **LangChain** and **LangGraph** frameworks. The first simplifies the integration of LLMs with external tools, APIs, and memory

modules, enabling the creation of flexible agents supporting a wide variety of models, from open-source to proprietary ones. Owing to this flexibility, the following chapter is dedicated to experimentation across multiple LLMs to determine which is most effective in this context. The current implementation employs **GPT-4o with vision**.
LangGraph extends LangChain with a graph-based orchestration layer for managing stateful agent workflows, supporting multi-step reasoning, parallel tool execution, and control over the decision path.
To maintain context and enable multi-turn interactions, the agent uses a short-term memory implemented through LangChain. This component stores both user queries and tool outputs, providing continuity across the reasoning process. Memory ensures that intermediate results, such as detected lesions or previous classifications, can be reused in subsequent steps, reducing redundant computations and supporting coherent dialogue with the user.

As shown in the figure 2.1 the system integrates multiple specialized tools, each implemented as an independent module that communicates with the LLM through structured calls. The toolset includes:

- **Detection tool**, used to identify and segment multiple lesions in panoramic body images;
- **Classification tool**, trained on heterogeneous dermatology datasets and enhanced with a concept bottleneck model;
- **Saliency tool**, which applies gradient-based and perturbation-based methods to generate interpretable visual explanations.

The modularity of the system ensures that tools can be updated or replaced independently by simply defining a new class, specifying the tool's input/output formats and capabilities. The LLM can then learn its usage without additional training.
For each tool, two classes are defined: the first inherits from `BaseModel` and defines the input parameters that the LLM must specify in the tool call; the second extends the `BaseTool` class from Pydantic and contains the logic of the tool. In particular, it must override the `_run` method, which implements the actual execution logic and is invoked whenever the LLM calls the tool. In addition to the synchronous `_run` implementation, an asynchronous counterpart `_arun` is also defined. This method allows the tool to be invoked in asynchronous workflows, where non-blocking execution is required.

To guide the agent toward producing a clinically accurate, transparent and safe answers the **system prompt** was defined as follow:

```
You are an expert medical AI assistant capable of answering
medical questions and analyzing dermatological images with
```

```
reasoning similar to that of a clinician. Your responses must
always be careful, transparent, and should never be presented
as a definitive diagnosis. At the end of every answer, you must
explicitly remind the user that your interpretation does not
replace a professional medical evaluation and that consultation
with a qualified specialist is required. Use your own visual
reasoning to analyze the images and rely on available tools to
complement and strengthen your conclusions. You may perform
multiple tool calls, either in sequence or in parallel, to
achieve a more complete and reliable assessment. Always review
and critically evaluate the outputs of each tool, highlighting
their usefulness, consistency, and possible limitations. If
essential information is missing, you may ask one concise,
specific follow-up question to clarify before proceeding, but
never fabricate or assume data that has not been provided.
When you describe results, write in a detailed yet concise
way, offering a structured medical interpretation. Include
the primary finding or suspected pathology, specify whether the
lesion appears benign or potentially malignant, and indicate
a qualitative confidence level such as low, medium, or high.
Support your reasoning by referencing the key visual evidence
observable in the image, such as asymmetry, border irregularity,
color variation, diameter, or evolution (following the ABCDE
criteria), as well as any notable dermoscopic structures if
visible. Whenever appropriate, you may mention one or more
plausible differential diagnoses and indicate any concerning
or suspicious features that would justify further clinical
evaluation. Suggest appropriate next steps, such as dermoscopic
examination, biopsy, or consultation with a dermatologist,
making clear that these are recommendations for further
assessment, not diagnostic conclusions. Tool-generated images
are shown in a box to the right of your textual output. Never
include Markdown image tags or HTML image elements, and never
display any file system paths or URLs, including localhost or
temporary directories. When a tool generates an image, refer
to it only in descriptive form, for example: "Here is the
result: an annotated image highlighting the most significant
regions is shown on the right." Always sanitize tool outputs
before presenting them, removing any technical paths, URLs, or
internal filenames that might appear. Maintain a professional
and informative tone throughout your explanations, clearly
stating the limitations of the analysis and acknowledging any
uncertainty arising from image quality, occlusions, or model
reliability. If your confidence in a finding is low, express
that uncertainty explicitly and explain why. Conclude every
```

```
response with a clear disclaimer such as:  "This is not a
medical diagnosis.  Please consult a qualified dermatologist
for definitive evaluation."
```

The prompt initially describe the expected agent behavior in the communication with the user. It instructs the system how to use the tools and elaborate its output, and how they should be described in the answer, critically evaluating their reliability and limitations.
Furthermore, it enforces strict formatting constraints, requiring the model to describe generated images only in natural language. Finally, the prompt ensures medical safety and ethical compliance by mandating an explicit disclaimer at the end of every response, clearly stating that the output is not a medical diagnosis and that consultation with a qualified dermatologist is always necessary for definitive evaluation.

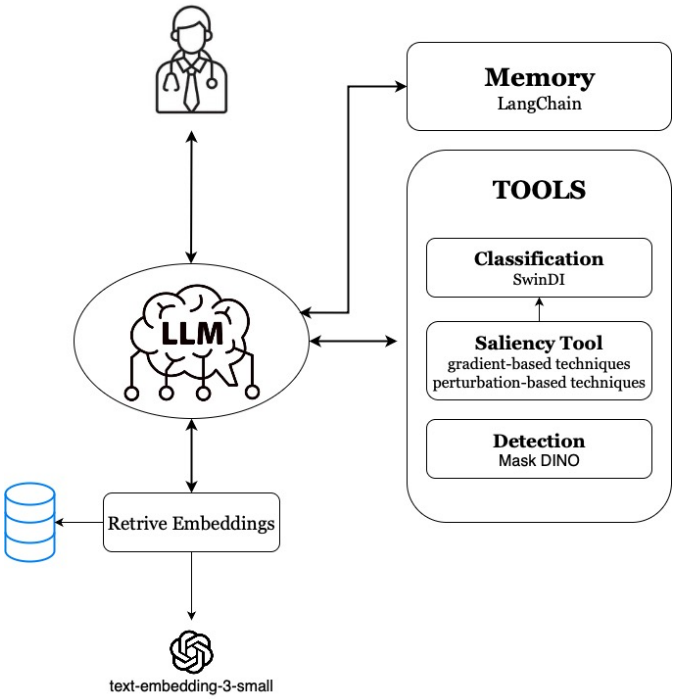


Figure 2.1: Overall design of the AI agent. The LLM acts as the central controller connecting the clinician's query to the different vision-based modules.

Finally, to ensure usability in clinical settings, the system has been equipped with a **Gradio-based interface**. The interface showed in the figure 2.2 enables users to:

- Upload dermatological images using the box on the right;

- Enter natural language queries in the text box at the bottom left;
- Receive interactive responses in the box on the left;
- View visual outputs displayed again in the box on the right;
- Clear the chat using the dedicated button.

Transparency is enhanced by showing intermediate tool results alongside the final reasoning, thereby increasing trust and usability in real-world scenarios.

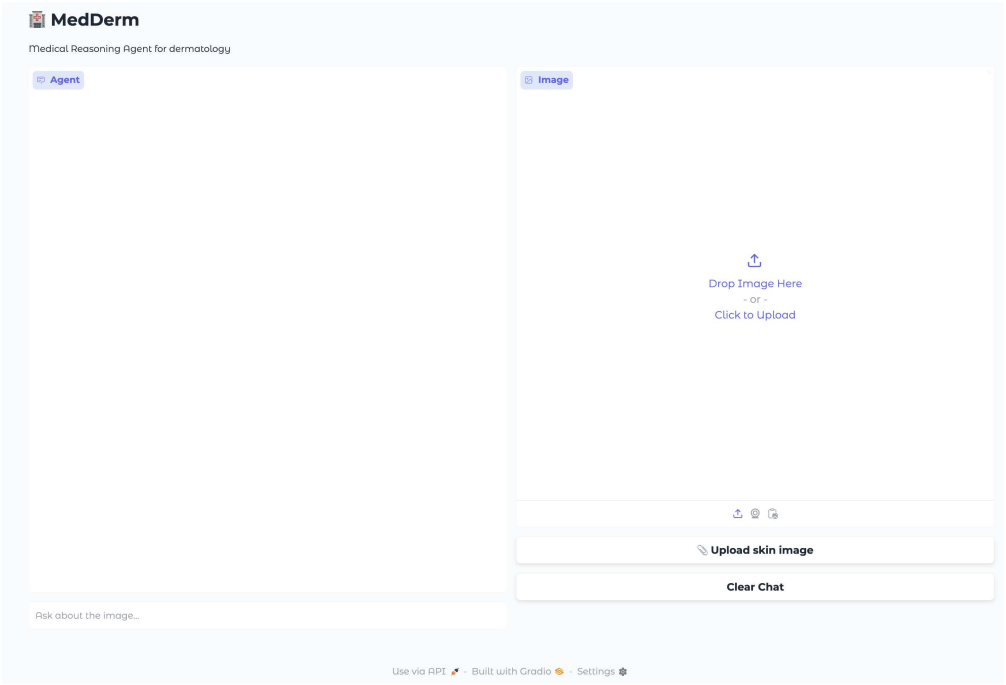


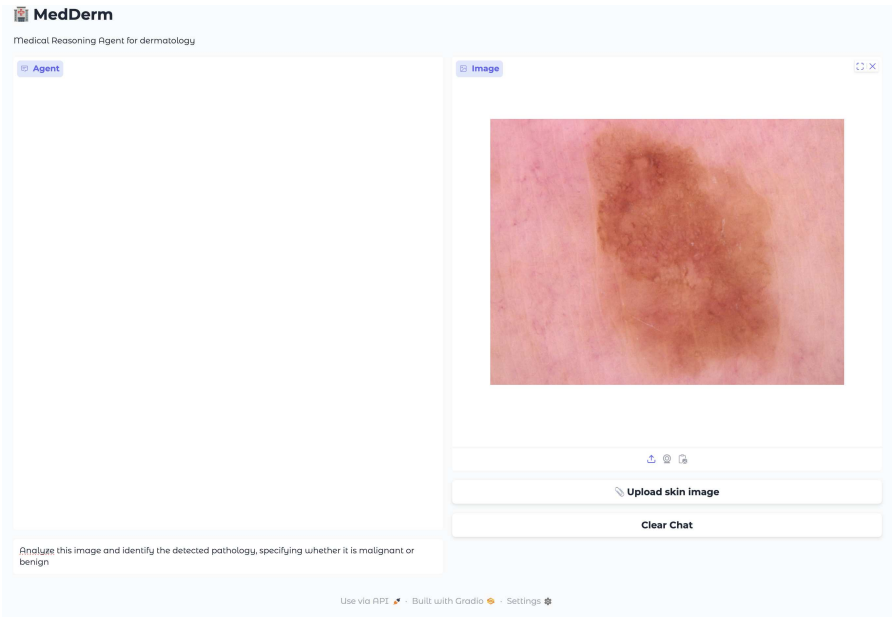
Figure 2.2: User interface. The layout displays a dual-panel structure, where the left section hosts the conversational agent for reasoning and dialogue, and the right section allows users to upload dermatological images and receive visual results from the system.

2.2 Classification Tool

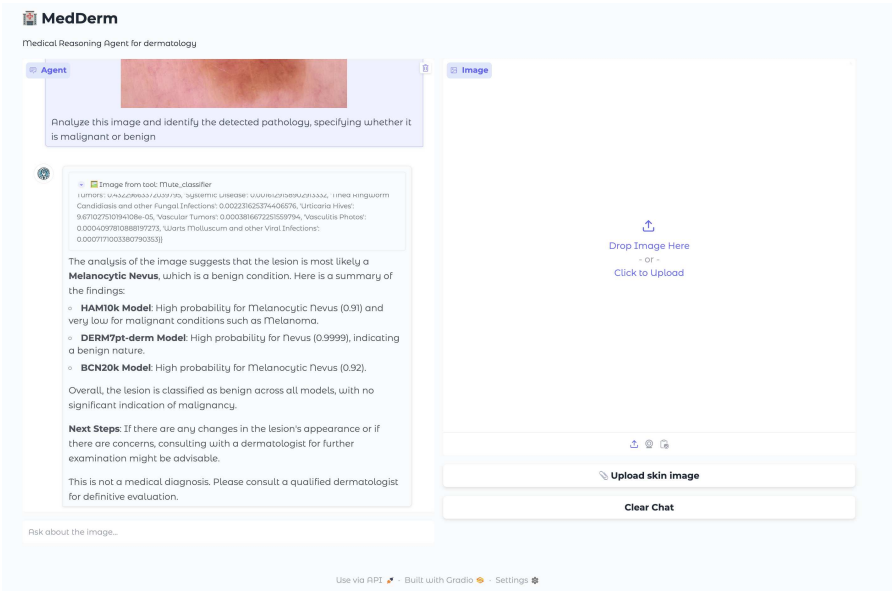
The classification tool is a central component of the proposed system. It is designed to classify dermatological images and provide reliable predictions across a wide range of conditions.
In this section, the structure of the model and the strategy adopted to handle the heterogeneity of available datasets in dermatology are presented.
The tool combines a powerful transformer-based backbone with architectural mechanisms that allow the integration of knowledge from heterogeneous label distributions, ensuring robustness and generalization. A description was defined for all the tools since it is crucial for guiding the agent in the correct selection of the tool to use. For the classification tool the description is the following one:

```
This tool analyses dermatologic images and classifies them
for multiple pathologies.  Input should be the path to a
dermatologic image file Output is a dictionary with heads (of
the model, directly connected to dataset used to train them)
as key and a dictionary of pathologies and their predicted
probabilities (0 to 1) as values.  Example:  {'fitzpatrick':
{'acanthosis nigricans':0.9987}} Reason over all the output
heads of the model in order to output only one pathology among
all of them
```

Figure 2.3 shows an example of how the system operates when the classification tool is employed. In the first image, the left panel allows the user to upload the dermatological image to be analyzed. The second image displays the agent's response, including the model predictions and the corresponding explanation of the classification results.



(a) Image upload interface



(b) Agent's classification response

Figure 2.3: Example of the system interface when the classification tool is used. (a) The right panel shows the interface section where the user can upload a dermatological image for analysis. (b) The left panel displays the agent's response, including model predictions and an explanation of the classification results.

2.2.1 Classification Model

The backbone of the model is a **Swin Transformer**, a hierarchical vision transformer that computes image features using shifted windows for self-attention.
Unlike standard Vision Transformers, which compute global attention across all image patches, the Swin Transformer restricts attention to local windows that are shifted between layers.
The hierarchical design further allows the extraction of multi-scale features, which is particularly beneficial for dermatological images where both local texture and global context are relevant.
On top of the backbone, the classification tool implements a multi-head architecture to deal with the heterogeneity of dataset labels.
Each dataset used in the training is associated with a dedicated head, trained independently on its respective label space.
This design prevents conflicts between incompatible taxonomies while preserving dataset-specific knowledge.
During inference, the features extracted by the Swin backbone are forwarded to all classification heads, since it is not possible to determine in advance which dataset a general dermatological image most closely resembles.
The output of the tool is a JSON object, where each key corresponds to a dataset name and the associated value represents the output of the corresponding head, expressed in terms of class probabilities. In the next chapter, an experiment will be presented to validate this approach, demonstrating the ability of the central LLM to reason across all output heads and improve the accuracy of individual heads by leveraging the results of the others.

2.2.2 Datasets

The model was trained on five widely used dermatological datasets, each contributing to generalization across different image types, skin tones, and clinical settings:

- **HAM10000** [39]: A large dataset of dermatoscopic images annotated into seven diagnostic categories, widely used as a benchmark in dermatology.
- **Derm7pt** [40]: Includes both dermatoscopic and clinical images, annotated according to the seven-point checklist.
- **BCN20000** [41]: A large-scale collection of dermoscopic lesion images acquired in real-world clinical settings, improving robustness to acquisition variability.
- **DermNet** [42]: A dataset of clinical photographs covering a wide range of dermatological conditions.

- **Fitzpatrick17k** [43]: A dataset with 114 skin conditions and diverse skin tones, ensuring inclusivity and improved generalization across patient populations.

The combination of these datasets allows the model to be more adaptable to different clinical contexts.

2.2.3 Concept Bottleneck Model

To enhance transparency and interpretability, the classification tool was further extended with a concept bottleneck model trained on the **SKINCON** dataset [44]. SKINCON contains images annotated by dermatologists with 48 clinical concepts, such as color, texture, and morphological patterns.

As shown in Figure 2.4, in this setup, the model is trained to predict these intermediate clinical concepts before producing the final diagnostic label. This forces the network to align its internal representation with human-understandable features.
At inference time, the tool outputs not only the predicted pathology but also the set of activated concepts that contributed to the decision.

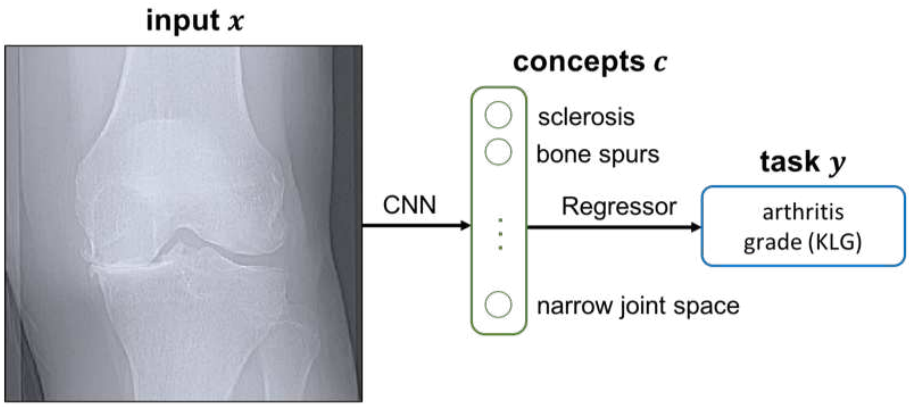


Figure 2.4: Example of a concept bottleneck model. Raw input images (medical X-ray and bird photo) are first mapped into high-level semantic concepts, which are then used to predict the final classification. [1]

However, since the number of images annotated at the concept level is limited, the overall performance of this component remains restricted. The current implementation should therefore be regarded primarily as a proof of concept, demonstrating the potential benefits that could be achieved if a larger quantity of concept-level annotated data were available.

Beyond improving interpretability, this approach also enables **concept intervention**. By exploiting the interactive nature of the agent, domain experts can

directly inspect and even modify the activated concepts through the conversational interface. This makes it possible to correct mispredicted concepts and guide the system towards more clinically meaningful predictions.

Figure 2.5 illustrates a possible workflow of the interaction between the agent and the dermatologist.

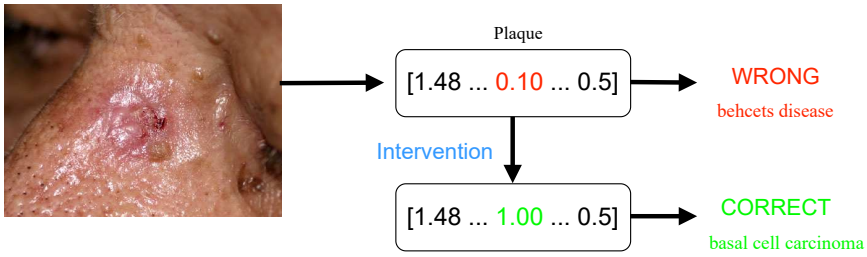


Figure 2.5: Concept Intervention: Example of how modifying an interpretable concept can alter the model's prediction. Increasing the activation of the Plaque concept changes the classification from Basal Cell Carcinoma to Behçet's Disease, illustrating the causal influence of semantic concepts on the model's decision.

2.3 Embedding Tool

The embedding tool is responsible for integrating patient metadata into the classification pipeline, ensuring that non-visual information such as age, sex, and lesion localization contributes to the diagnostic reasoning process. While dermatological images provide essential visual cues, contextual metadata can significantly influence classification accuracy and alignment with real-world clinical practice. According to consultation with an expert dermatologist, the most relevant patient information for this task are: age, since some conditions are more frequent in older patients; sex; and the localization of the lesion on the body.

To integrate this information into the reasoning process, patient metadata must be brought into the same feature space as the image representations. This is achieved by leveraging text embeddings.

Text embeddings are dense numerical representations of natural language that capture semantic information in a continuous vector space. The goal of an embedding model is to map semantically similar inputs (e.g., "male patient" and "the patient is a man") to vectors that are close to each other, while semantically unrelated inputs are mapped to distant vectors.

After several experiments with different text-embedding models, the one that achieved the best performance was OpenAI's *text-embedding-3-small* model. This state-of-the-art embedding generator is designed for a broad range of applications,

including classification, semantic search, clustering, and recommendation tasks. It processes natural language input and projects it into a high-dimensional embedding space; in this case, using the *text-embedding-3-small* model, the embedding is represented as a vector of 1536 real numbers. The model was accessed via the OpenAI API[1], which provides official support and documentation for embedding generation.
To leverage text embeddings in the dermatological classification pipeline, structured metadata must first be transformed into natural language sentences. This step is crucial, as the embedding model is trained to operate on free text rather than categorical variables.

For example:

- **Metadata:** { "age": 45, "sex": "female", "localization": "back" }
- **Transformed input:** "The patient is 45 years old, who is female, with a skin lesion located on the back."

By converting structured metadata into natural language descriptions, the embedding model can exploit its semantic capabilities to generate meaningful vectors that encode the available patient information. The retrieval of patient metadata can be performed in two ways:

- The user manually enters them through the interface.
- The patient information is automatically queried from a database.

Once generated, the metadata embeddings are projected to the same dimensionality as the visual features extracted by the Swin Transformer. As shown in Figure 2.6, the embeddings are then combined, through a simple summation operation, with the pooled features of the classification model. This fusion allows the final classifier heads to reason jointly over image features and patient context. For example, the same lesion appearance might suggest different probabilities depending on whether the patient is a child or an adult, or whether the lesion is localized on the face or on the hands. As shown in the experimental section, this approach leads to an improvement in the overall performance of the classification model.

[1] `https://platform.openai.com/docs/guides/embeddings`

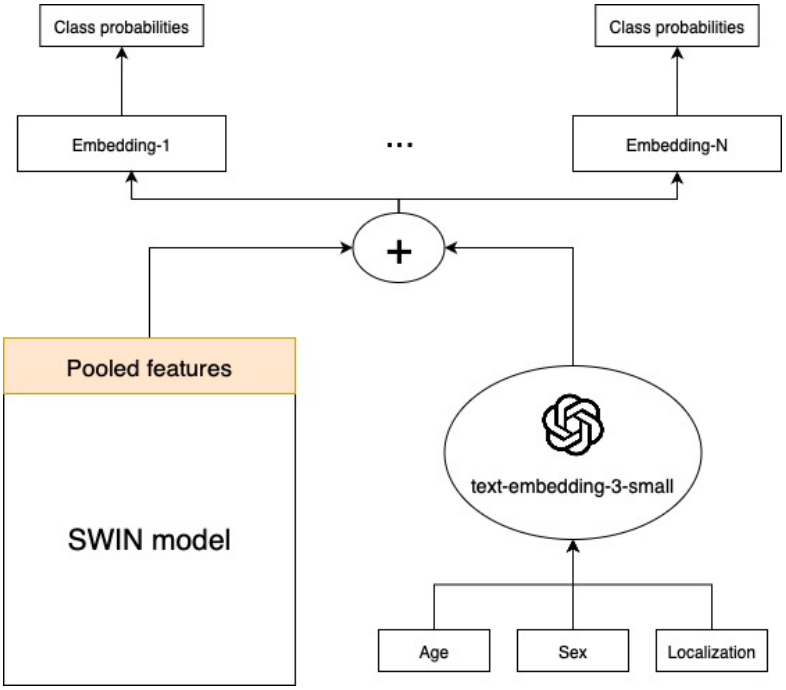


Figure 2.6: Architecture of the embedding tool. Patient metadata is converted into natural language, embedded into a numerical vector using the *text-embedding-3-small* model, projected to the same dimensionality as the visual features, and combined with the Swin Transformer representations.

2.4 Saliency Tool

The saliency tool provides a visual explanation of the classification results, highlighting which regions of the input image most strongly contributed to the prediction. This component is fundamental in the context of explainable AI (XAI), as it allows clinicians to verify whether the system's reasoning is coherent with medical practice and focused on clinically relevant areas of the skin.

Among the tested methodologies, the approach that achieved the most reliable and interpretable results is **Vanilla Gradient**.
This method belongs to the family of gradient-based explainable techniques, and it is the simplest one because it just computes the importance of each input feature by directly evaluating the gradient of the model's output with respect to the input pixels. In the context of our agent, this results in a saliency map where the most influential pixels are highlighted according to their impact on the target prediction. The tool was implemented using the **Captum** library.

The overall workflow is coordinated by the central agent, which can call the tool by specifying the path of the image and the **target label** to analyze. This image is

not an endpoint in itself: it is passed back to the central LLM, which integrates the visual evidence into its prompt, supporting the reasoning chain in such a way that it can engage in a clinically oriented discussion with the dermatologist, grounding its output in interpretable visual cues.

Generally, the agent is instructed to use as target label the predicted class of the image classifier, but it is possible that reasoning over multiple saliency maps generated for different target labels could enhance the agent's reasoning process. For this reason, in cases where the output of the classification tool is not confident about a specific class, the saliency tool may be called more than once for the same image. To instruct the central LLM to use the tool correctly the following instruction prompt was defined:

```
A tool that generates a saliency map for a given image to
explain the classification model's prediction.  Before calling
this tool, you must have already classified the image using the
MuteClassifierTool.  Pay attention to use the same head used
for the classification.  And pay attention to the spelling of
the target class, it must be exactly the same as in the model's
classes.  It takes an image path, and return the saliency maps,
the generated pictures represent in the left the original
image in the centre the image masking and in the right the
blended heatmap, that highlights the regions of the image that
contributed most to the prediction.
```

To enhance interpretability, the attributions are visualized through three complementary modalities:

1. **Original Image**: To preserve the clinical context.
2. **Masked Image**: To highlight which regions were considered important or irrelevant by the model.
3. **Blended Heat Map**: To overlay the attribution map on the original image, providing an intuitive visualization of the salient areas.

An example of the explanation is shown in Figure 2.7, where it can be seen how easily the user can simultaneously assess the raw input and the isolated relevant features. As a continuation of the chat showed in figure 2.3, in this other image 2.8 it is asked to the agent to produce the saliency map of the lesion under analysis. On the right side, it is showed the final result integrated in the left with the agent analysis.

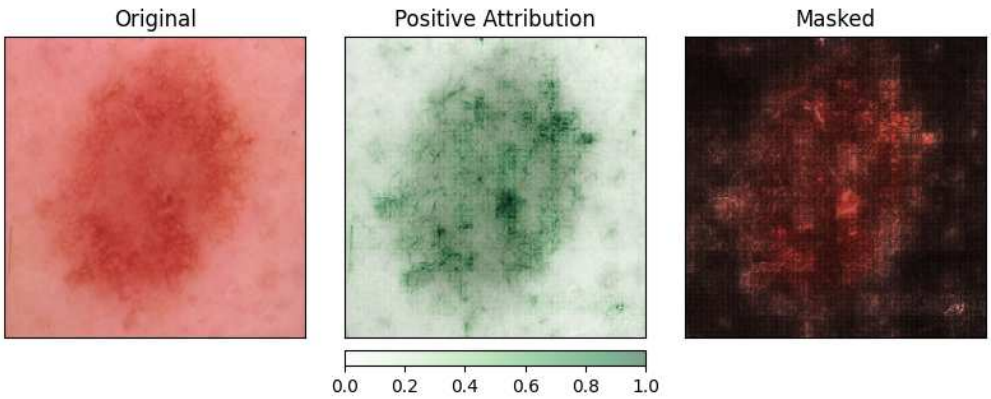


Figure 2.7: Example of Vanilla Gradient-based explanation. The figure shows the original image, the saliency mask, and the blended heat map highlighting the regions most influential in the classification decision.

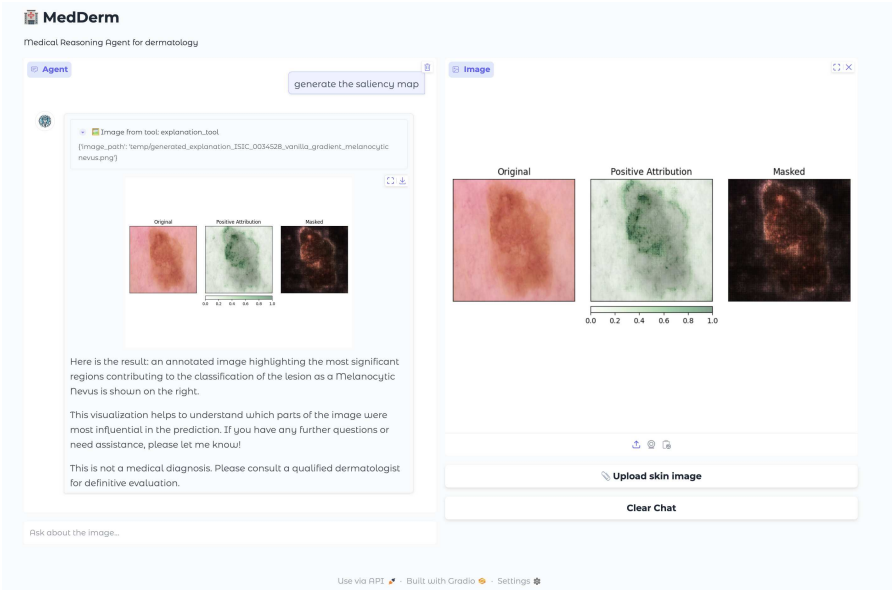


Figure 2.8: Example of the interface during the explanation phase. On the right, the visualization displays the original image, the positive attribution map, and the masked result, illustrating how the explanation tool enhances interpretability by linking model predictions to visible clinical features.

2.5 Detection Tool

The detection tool is responsible for identifying and localizing multiple suspicious lesions in wide-field dermatological images. For this purpose, the system leverages **MaskDINO**, a state-of-the-art transformer-based model for object detection and instance segmentation. This model has demonstrated high performance in generating both bounding boxes and segmentation masks, making it particularly suitable for the analysis of panoramic body images where several lesions may coexist.

The model was trained on the **SKINPAN dataset**, a large collection of 10,050 high-resolution panoramic dermatology images specifically created to support research into lesion detection and segmentation in realistic clinical scenarios. The dataset was developed at the *University of Insubria* by PhD student **Mattia Gatti**, and represents one of the first resources explicitly focused on panoramic clinical imagery.

Unlike traditional dermatoscopic datasets such as HAM10000 or BCN20000, which mostly include close-up images of single lesions, SKINPAN reflects the conditions of real clinical practice. Each image depicts broad anatomical regions (e.g., back, chest, abdomen) and contains one or more lesions selected by expert dermatologists for observation. This context is crucial for diagnostic workflows, as clinicians often evaluate the distribution, symmetry, and relative characteristics of multiple lesions within the same patient.

Data collection took place at the *Hospital Circolo e Fondazione Macchi di Varese* between 2014 and 2025. Panoramic images were captured by board-certified dermatologists using the **FotoFinder Universe** system and the **Medicam 800HD** camera, ensuring standardized resolution (1920×1080), lighting, and patient positioning. Each image was marked with red arrows indicating suspicious lesions, which served as ground truth for the annotation phase.

The annotations were performed through a semi-automated pipeline that combined the **Segment Anything Model (SAM)** with manual refinement. The red arrows placed during acquisition were automatically extracted and used as prompts for SAM. Each lesion was labeled as *"selected for observation"*, indicating that, while not necessarily malignant, it was clinically relevant and required further analysis.

The dataset includes both bounding boxes and segmentation masks for all lesions in **COCO format**. Moreover, metadata such as patient age and anatomical site of the lesion are also available.

An example of images belonging to the dataset with the relative ground truth is shown in Figure 2.9.

Additionally, to compensate for the lack of negative samples, the dataset incorporated synthetic inpainting using **Stable Diffusion XL (SDXL)**, producing lesion-free versions of selected images.

The integration of MaskDINO with the SKINPAN dataset provides a robust detection tool capable of analyzing wide-field dermatological images in a clinically meaningful way. The tool is called by the central LLM when it recognizes that the image contains more than one lesion to be analyzed. The following instruction prompt is used to guide the LLM in the correct call of this tool:

```
The tool detects dermatologic lesions in panoramic images
using Mask DINO. The tool must be called only if the image
is panoramic so if in the image are present more that one
dermatological lesion.  Returns bounding boxes, scores, and
the path to an image with boxes drawn.  You don't need to give
further details about the boxes, it is enough to show the image
with the boxes drawn on it.
```

Figure 2.10 illustrates the workflow used to interact with the agent when analyzing panoramic images.

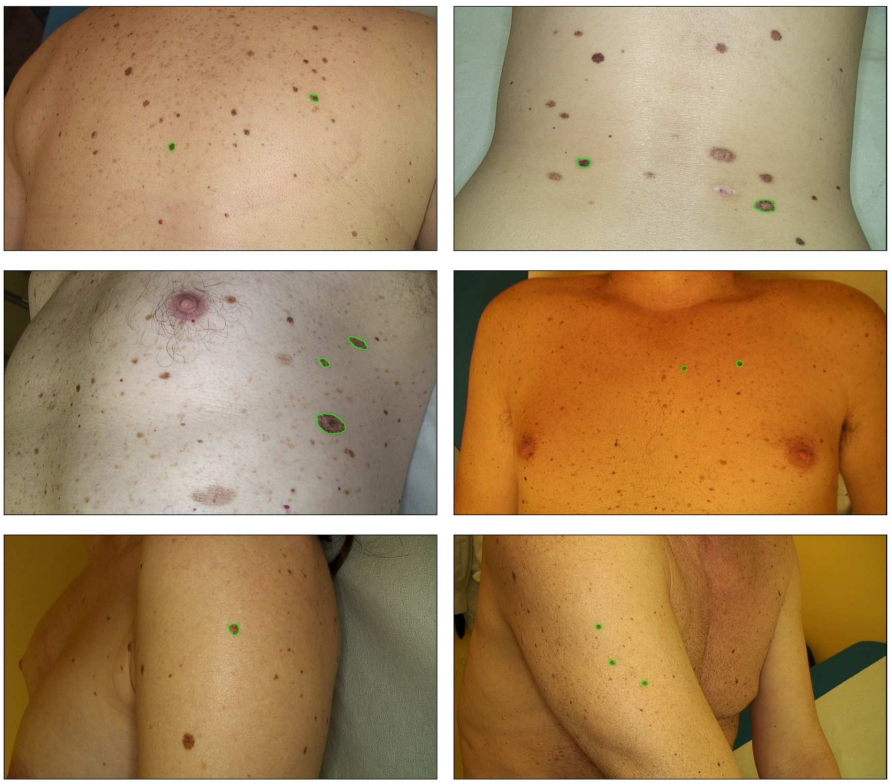
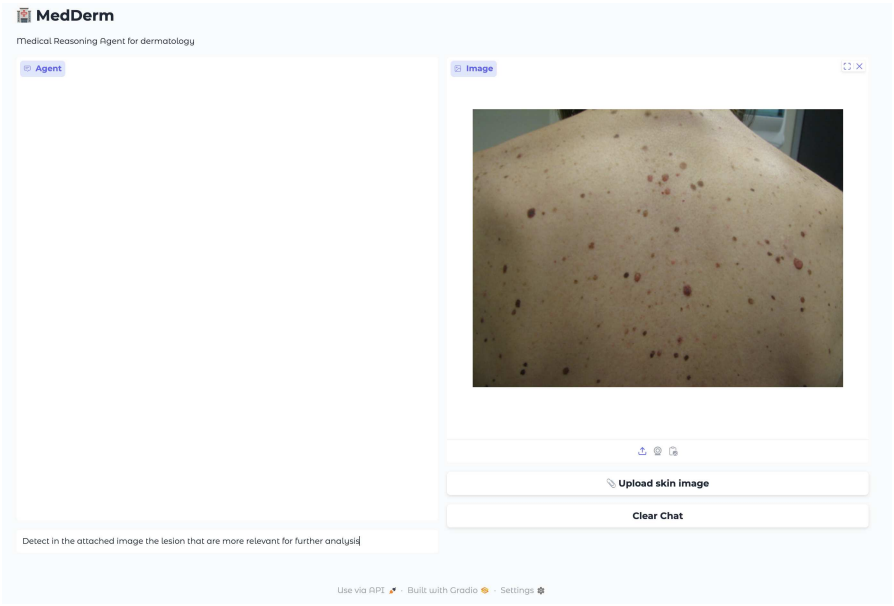
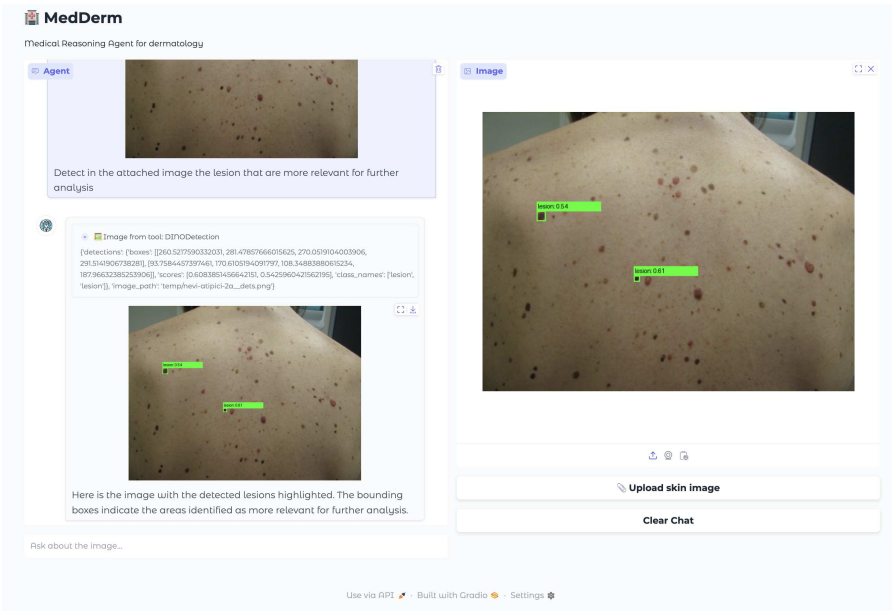


Figure 2.9: Examples of images from the SKINPAN dataset with annotated lesions (green contours). The dataset provides bounding boxes and segmentation masks created through a semi-automated pipeline.



(a) Image upload interface



(b) Agent's detection response

Figure 2.10: Example of the system interface when the Detection tool is used. (a) The right panel shows the interface section where the user can upload a panoramic image for analysis. (b) The left panel displays the agent's response, while on the right panel the final image with the predicted bounding boxes

Chapter 3

Experiments

This chapter presents the experimental evaluation of the proposed framework. The objective is to systematically assess the contribution of each module to the overall performance of the dermatological AI agent, analyzing not only accuracy but also robustness and interpretability. The experiments are organized into four main sections, each focusing on one of the core components of the system:

- **Reasoning Core – Large Language Model**: This part evaluates the ability of state-of-the-art multimodal LLMs, specifically GPT-4o, Gemini 2.0 Flash, and Gemini 2.5 Flash, to reason over dermatological images and interact with the classification tool (SwinDI).

- **Text Embeddings for Metadata**: This section investigates how patient information (age, sex, localization) can be effectively integrated into the classification pipeline. Different strategies for metadata embedding are tested to measure their impact on classification accuracy.

- **Concept Bottleneck Model**: Here the focus is on evaluating the integration of concept-level annotations from the SKINCON dataset. The experiments explore how the prediction of intermediate clinical concepts improves interpretability, while also highlighting the limitations imposed by the relatively small number of concept-annotated samples.

- **Explainable AI Techniques**: Finally, this section examines the interpretability of the model's predictions. Several saliency-based methods are compared in terms of visual coherence and robustness when the target class changes.

Together, these experiments highlight the strengths of the modular design by testing each tool in isolation and observing their combined effect on the overall system.

3.1 Reasoning Core – Large Language Model

In this section, we evaluate the reasoning capabilities of multimodal LLMs in the dermatological context. Three models were tested: **GPT-4o**, **Gemini 2.0 Flash**, and **Gemini 2.5 Flash**.
All experiments were conducted using the HAM10000 dataset for dermatological images, complemented with 1,000 non-medical images from ImageNet [45] for the binary classification tasks.
The tool integrated in these experiments was exclusively the SwinDI classifier, while other modules of the agent were not considered.

A significant challenge in using a vision model as a tool for a multimodal LLM lies in the fact that images are represented solely as features. To address this, we considered two integration methods for incorporating images into the LLM's pipeline. In the first, the image path is passed as a text input to the LLM, which then uses this path to access the image when invoking the vision tool. In the second, the integration of the image into the prompt of the LLM is achieved by uploading the image in `base64` format and passing it in the API call, as shown in the following code snippet:

```
with open(image_path, "rb") as img_file:
    img_base64 = base64.b64encode(img_file.read()).decode("utf-8")

messages.append({"role": "user", "content": f"the image is located at: {image_path}"})
messages.append(
    {
        "role": "user",
        "content": [
            {
                "type": "image_url",
                "image_url": {"url": f"{img_base64}"},
            }
        ],
    }
)
```

3.1.1 Experiment Setup

The experiments fall into three main categories:

- **Binary classification (dermatology vs non-dermatology):** this experiment tests whether the LLM can distinguish dermatological images from

non-medical ones. The experiment is carried out in two variants:

  - **Direct answering** ("YES/NO") without tool calls.
  - **Tool-call evaluation**: The challenge is to call the tool only when the LLM analyzes a dermatological image, and avoid wasting time when the image does not contain a lesion. If the LLM decides to invoke SwinDI, it must also provide a short textual explanation of the image.

  The instruction prompts used are the following:

  - For the binary classification without tool:

    ```
    If the image contains a skin lesion answer YES, otherwise
    NO. Output [YES, NO] only, without any additional text or
    explanation.
    ```

  - For the binary classification with tool:

    ```
    If the image refers to skin lesions, answer 'Yes' and
    perform the classification with the tools and use this
    information added to the initial text to answer the
    question in detail and explain the characteristics
    observed in the image.
    ```

- **Multi-class classification without tool:** The goal is to evaluate the inherent visual ability of the LLM. The task is to classify the images of the HAM10000 test set into seven categories (MEL, NV, BCC, AKIEC, BKL, DF, VASC) based only on its internal multimodal capacities, without tool support. The instruction prompt is the following:

  ```
  Classify this image, using your own vision and reasoning,
  into one of the seven classes:  MEL, NV, BCC, AKIEC, BKL,
  DF, VASC. Output the class name only, without any additional
  text or explanation.
  ```

- **Multi-class classification with tool:** Evaluation of how the LLMs integrate the tool output of SwinDI. SwinDI produces predictions from all dataset-specific heads. The LLM must reason over these outputs together with its own vision to provide the final classification. In this experiment the following instruction prompt is used:

  ```
  Classify this image, using your own vision and reasoning
  into one of the seven classes:  MEL, NV, BCC, AKIEC, BKL, DF,
  VASC. Use tools to complement your reasoning.  Critically
  think about the tool outputs.  Output the class name only,
  without any additional text or explanation.
  ```

3.1.2 Results

Table 3.1: Binary classification accuracy (skin lesion vs. no skin lesion) on images from HAMTest1512 and a subset of ImageNet. Performance comparison across multimodal LLMs using either direct question answering or Tool-call evaluation approach.

Model	Direct	Tool called
GPT-4o	99.92%	**99.96**%
Gemini 2.0 Flash	99.72%	97.25%
Gemini 2.5 Flash	99.20%	99.68%

The binary classification task reflects a more general diagnostic triage, such as deciding whether a specialized dermatology tool is needed. Multimodal LLMs appear to perform significantly better in this simplified context. Table 3.1 reports the classification accuracies on this binary task using images from **HAMTest1512** and a subset of **ImageNet**.
GPT-4o demonstrates exceptional performance, reaching 99.92% accuracy when directly answering the binary question. Its performance improves slightly when tools are used (99.96%), suggesting that tool usage can marginally enhance reliability even in straightforward tasks. Similarly Gemini-2.0-Flash and Gemini-2.5-Flash perform strongly. The slightly drop in performance may indicate less efficient tool invocation compared to GPT-4o, but overall performance remains robust.

Table 3.2: Comparison of full classification performance on HAMTest1512 between Vision Models and Multimodal Language Models.

Model	Accuracy
SwinDI	**86.64**%
GPT-4o	52.71%
Gemini 2.0 Flash	52.05%
Gemini 2.5 Flash	28.84%

As shown in Table 3.2, SwinDI achieves the highest accuracy at 86.64%, substantially outperforming all other tested models. The accuracy of the model is obtained by looking at the specific HAM10000 head. This result confirms the strong suitability of domain-specific vision models for fine-grained dermatological classification tasks.

In contrast, all tested multimodal LLMs perform significantly worse. GPT-4o, the best among them, reaches only 52.71%, which is almost 34 percentage points

below SwinDI. This sharp gap underscores a critical limitation of general-purpose multimodal LLMs: despite their broad capabilities in vision-language reasoning, they lack the fine-grained discriminative power needed for medical image analysis without the support of dedicated vision tools.
Gemini 2.0 Flash performs at a similar level, with 52.05%, confirming that even across different architectures, general-purpose LLMs struggle to achieve clinically meaningful accuracy in this highly specialized task. Gemini 2.5 Flash shows a severe performance drop, reaching only 28.84%, which highlights instability and inconsistency when applying generalist models directly to medical domains.

Table 3.3: Comparison of full classification performance on HAMTest1512 between Multimodal Language Models, using the tool output.

Model	Accuracy
GPT-4o	**86.77%**
Gemini 2.0 Flash	76.32%
Gemini 2.5 Flash	86.18%

As shown in Table 3.3, the integration of the SwinDI classification tool radically changes the performance. All three multimodal LLMs significantly improve their accuracy when reasoning over the tool's outputs, demonstrating the effectiveness of combining a domain-specific vision model with a general-purpose LLM.

GPT-4o achieves the best result at 86.77%, even surpassing the performance of SwinDI alone. In four cases where the tool's output was uncertain, the LLM adjusted the prediction, and in two of these cases it selected the correct class. This shows that GPT-4o is not only capable of orchestrating tool calls but also of integrating external evidence into coherent and accurate reasoning.

Gemini 2.0 Flash reaches 76.32% accuracy. However, its performance remains below the other models because, in this experiment, the LLM did not invoke the tool in almost **400 cases**, relying instead on its own visual capabilities which, as shown in the previous experiment, are limited.

This phenomenon is not observed in **Gemini 2.5 Flash**, which in most cases relied on the tool's output. Nevertheless, in 61 cases it overrode the tool's prediction, but less than half of these changes were correct. Despite this, its overall performance remains strong, achieving **86.18%** accuracy.

3.2 Embedding Tool - Text Embeddings for Metadata

The integration of patient metadata into the classification pipeline aims to enrich the diagnostic process with contextual information that may strongly influence the prediction. Through consultation with an expert dermatologist, three attributes were selected from the available metadata as most relevant for dermatological classification: **age**, **sex**, and **lesion localization**. These features were chosen because certain pathologies are more prevalent in specific age groups, show different distributions across sexes, or tend to appear in particular anatomical regions.

3.2.1 Datasets

All experiments were conducted on the HAM10000[39] and BCN20000[41] datasets, using the classification head associated with the respective dataset. For all experiments, the models were trained from scratch using the same set of training hyperparameters. The datasets were consistently divided into *training*, *validation*, and *test* sets, maintaining identical splits across experiments to ensure a fair comparison.

3.2.2 Experiment Setup

The training configuration is reported below:

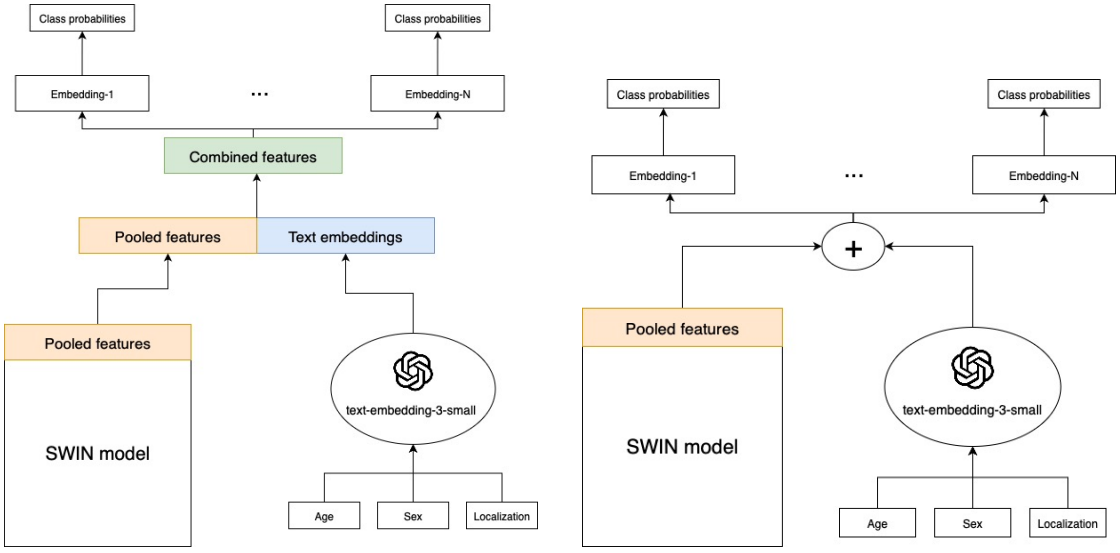
- Batch size: 24
- Number of epochs: 100
- Learning rate: 5×10^{-5}
- Optimizer: Adam
- Early stopping applied if the validation accuracy did not improve for more than 10 consecutive epochs

The first investigation aimed to establish the most effective method to integrate textual embeddings with visual features extracted by the Swin Transformer classifier. Two strategies were tested, both evaluated on OpenAI's text-embedding-3-small model on the HAM10000 dataset:

- **Concatenation approach**: The embedding vector representing the metadata was concatenated with the image feature vector produced by the Swin Transformer. This combined vector was then projected back onto the image feature dimensionality before being passed to the classification heads.

- **Summation approach**: The embedding vector was projected onto the same dimensionality as the Swin feature vector and then simply added element-wise.

Figure 3.1 shows a visual explanation of the difference between the two approaches.



(a) Concatenation approach (b) Summation approach

Figure 3.1: Comparison of two strategies for integrating metadata embeddings with Swin features. (a) Concatenation of pooled image features with text embeddings, followed by projection. (b) Projection of text embeddings to the same dimension as pooled image features and element-wise summation.

Table 3.4: Comparison of full classification with text-embeddings metadata on HAMTest1512 between two approaches: Concatenation approach, Summation approach

Approach	Accuracy
Concatenation	87.09%
Summation	**88.28%**

As shown in Table 3.4, both integration strategies for metadata embeddings lead to an improvement over the baseline performance of 86.64% (without metadata). **The concatenation approach** reaches 87.09%, providing only a marginal gain of less than half a percentage point, suggesting that this integration approach does not fully exploit the information provided by metadata. In contrast, the summation

approach achieves 88.28%, corresponding to a clear improvement of more than 1.6 percentage points over the baseline.
This result indicates that simply summing element-wise the projected embeddings allows for a more natural and effective fusion of contextual information with visual features. As a result, all subsequent experiments were conducted using this summation strategy.

Once the integration method was fixed, the next step was to compare different text embedding models available via public APIs. Two models were tested: **OpenAI text-embedding-3-small** and **Google Gemini text-embedding-004**. The goal was to evaluate which model provides embeddings that best complement the visual features in dermatological classification tasks. For each experiment, metadata was first transformed into natural language sentences (e.g. "The patient is 45 years old, who is female, with a skin lesion located on the back"), then encoded into dense numerical vectors by the embedding model, projected, and finally summed with the Swin features.

3.2.3 Results

Table 3.5: Test accuracy (%) on HAM10k and BCN20k of the SwinDI classification model with the integration of patient metadata through the text embedding from OpenAI text-embedding-3-small and Google Gemini text-embedding-004.

Model	HAM10k Acc.	BCN20k Acc.
OpenAI text-embedding-3-small	**88.28%**	74.85%
Gemini text-embedding-004	86.43%	74.85%

As shown in Table 3.5, the integration of patient metadata through text embeddings produces measurable improvements compared to the baseline performance of the SwinDI classifier without metadata, which achieves 86.64% on HAM10k and 74.39% on BCN20k. As discussed in the previous experiment, for HAM10000 the use of OpenAI's text-embedding-3-small increases accuracy to 88.28%, corresponding to a gain of approximately 1.6 percentage points over the baseline.
This demonstrates that contextual information such as age, sex, and lesion localization can effectively complement visual features, improving diagnostic robustness in a dataset where intra-class variability is high. The Gemini text-embedding-004 also contributes positively, reaching 86.43%, though its performance remains close to the baseline, suggesting that its embedding space may be less aligned with the dermatological context compared to OpenAI's model.

For BCN20000, both embedding models reach 74.85%, improving slightly over the baseline. The more modest gain observed in this dataset may be due to its higher heterogeneity and noise, which reduce the impact of metadata relative to visual features.

Overall, these results confirm the importance of metadata integration in dermatological AI. The proposed strategy allows the classifier to exploit complementary information, increasing performance and opening the possibility for future development of Explainable AI techniques over the metadata to further enhance the transparency of the AI Agent.

3.3 Classification Tool - CBM

The Concept Bottleneck Model (CBM) aims to improve interpretability by introducing an intermediate layer of clinically meaningful concepts between the image features and the final pathology classification.

3.3.1 Training Strategies

Two different training strategies were evaluated:

- **Independent training**: In this configuration, the model is divided into two phases. First, a classifier is trained to predict the set of clinical concepts from the input image. Then, a second classifier is trained independently to predict the final pathology label, using as input the *ground-truth concepts* instead of the predicted ones. This approach allows the pathology classifier to learn under ideal conditions, but at inference time it cannot rely on perfect concept annotations, thus introducing a performance gap.

- **Joint**: In this configuration, the entire model is trained entirely. Both the concept predictor and the pathology classifier are optimized simultaneously, using a combined loss function that balances the concept prediction loss and the final classification loss. The relative importance of the two terms is controlled by a weighting parameter λ, allowing different trade-offs between concept accuracy and task accuracy.

3.3.2 Dataset

All the experiement were conducted on the **SKINCON** dataset [44], which contains dermatological images annotated with 48 expert-defined clinical concepts (e.g. color texture and morphological patterns).

3.3.3 Metrics

Two different metrics were adopted for evaluation:

- **Concept prediction:** Since concepts classification is a multi-label problem, the **Area Under the Receiver Operating Characteristic curve (AUROC)** was used. The ROC curve is a graphically tool used to evaluate the performance of a binary classifier across different decision thresholds from 0 to 1. on the x-axis we find the False Positive Rate (FPR)

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

  on the y-axis the True Positive Rate (TPR) also called sensitivity or recall.

$$\text{TPR} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

  The AUROC is calculated as the area under this curve for each of the 48 clinical concepts treated as binary classifier where the concept under analysis is considered the positive class while the others are the negative ones. Finally the average is calculated to obtain the final value. An AUROC of 1 indicates perfect discrimination, while 0.5 corresponds to random guessing.

- **Task prediction:** the final pathology classification task was evaluated using **accuracy**, defined as the proportion of correctly classified images over the total number of test images.

3.3.4 Loss Functions

Two types of loss function were employed:

- **Concept prediction: Binary Cross-Entropy (BCE)**: For a multilabel setting with C concepts, the BCE loss is defined as:

$$\mathcal{L}_{BCE} = -\frac{1}{C}\sum_{c=1}^{C}\left[y_c \log(\hat{y}_c) + (1 - y_c)\log(1 - \hat{y}_c)\right]$$

  where $y_c \in \{0,1\}$ is the ground-truth label for concept c and $\hat{y}_c \in [0,1]$ is the predicted probability.

- **Final classification: Cross-Entropy (CE)**: For a multiclass classification problem with K classes, the CE loss is defined as:

$$\mathcal{L}_{CE} = -\sum_{k=1}^{K} y_k \log(\hat{y}_k)$$

where y_k is the one-hot encoded ground-truth label and $\hat{y}_k$ is the predicted probability for class k.

For joint training, the combined loss is:

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{BCE}$$

3.3.5 Results

The results of the experiment are reported in Table 3.6

Table 3.6: Results of the Concept Bottleneck Model on the SKINCON test set using independent and joint training with different λ values.

Training strategy	Concept AUROC	Task Accuracy
Independent	**0.815**	0.149
Joint, $\lambda = 0.001$	0.585	0.373
Joint, $\lambda = 0.01$	0.678	**0.391**
Joint, $\lambda = 0.1$	0.721	0.366
Joint, $\lambda = 1$	0.769	0.373

Figure 3.2 shows the trade-off between concept error (1 - AUROC) and task error (1 - Accuracy).

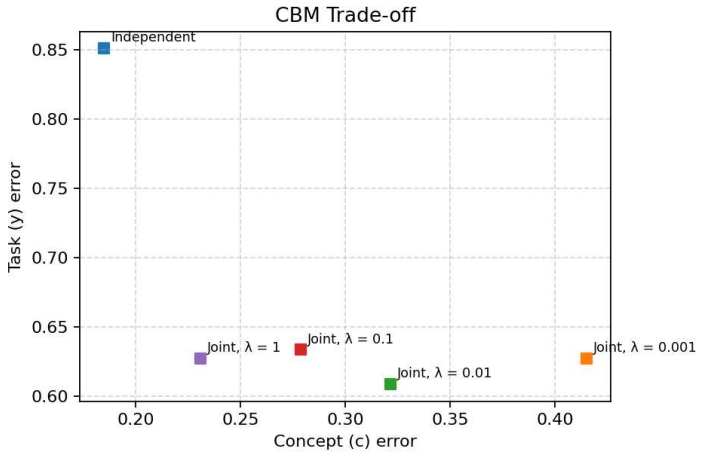


Figure 3.2: Trade-off between concept prediction and task prediction in the Concept Bottleneck Model. The x-axis represents the concept error (1 - AUROC) and the y-axis represents the task error (1 - Accuracy).

3.3.6 Discussion

The results highlight a clear trade-off between concept prediction and task accuracy: The **independent** training strategy achieves the highest AUROC for concept prediction (0.815), showing that the model can learn meaningful concept-level representations. However, task accuracy remains very low (0.149), as the pathology classifier is trained on ground-truth concepts and cannot handle the noisy predicted concepts at inference time.
The **joint** strategy sacrifices some concept accuracy in order to improve the final classification task. For example, with $\lambda = 0.01$ the model achieves the best task accuracy (0.391) despite a lower concept AUROC (0.678). This demonstrates that learning concepts and pathologies together can result in better end-to-end performance.
Increasing λ shifts the balance towards better concept learning but slightly reduces task accuracy. Conversely, very low λ values prioritize task prediction but degrade concept interpretability.
For a fair evaluation of the results, a further experiment was conducted. The original model (without the CBM architecture) was trained on the SKINCON dataset using the Fitzpatrick label distribution and was therefore evaluated on the SKINCON test-set only on the head corresponding to the Fitzpatrick dataset.
Table 3.7 shows a comparison between the task accuracy of the different training versions of the concept bottleneck model and the original model.

Table 3.7: Comparison between the task accuracy over the SKINCON test-set of the original model and the different training versions of the Concept Bottleneck Model (CBM).

Model Version	**Accuracy**
Original model	**0.4161**
Independent	0.149
Joint, $\lambda = 0.001$	0.373
Joint, $\lambda = 0.01$	0.391
Joint, $\lambda = 0.1$	0.366
Joint, $\lambda = 1$	0.373

The introduction of the Concept Bottleneck Model (CBM) leads to a slight deterioration in task accuracy compared to the original model. This behavior is expected, as the additional concept prediction layer constrains the model's capacity to optimize purely for performance. However, this trade-off is a typical and acceptable outcome: while the CBM slightly reduces accuracy, it significantly enhances interpretability, allowing the model's decisions to be better understood and grounded in human-interpretable clinical concepts.

3.4 Saliency Tool - Explainable AI Techniques

These experiments were conducted to perform a comparative analysis of different methodologies for generating saliency maps, with the aim of assessing their effectiveness in the dermatological context.
As explained in Section 1.3, saliency maps provide a visual explanation of the model predictions by highlighting the regions of the image that contributed most to the classification.
This type of explanation is particularly relevant in clinical practice, as it allows one to evaluate the alignment of the model's reasoning with the visual evidence typically used by dermatologists.

3.4.1 Experiment Setup

The experiments were implemented using **Captum**, an open-source library by Meta that provides a wide range of attribution algorithms for deep neural networks. Four explanation techniques were tested:

- **Vanilla Gradient**: Computes the gradient of the target class with respect to the input pixels. Based on the value of the gradients, the attribution matrix is generated and then overlaid on the original image to create the saliency map.

- **Input $\times$ Gradient**: Multiplies the gradients by the input values, giving more weight to pixels with higher intensity. This method emphasizes the relative contribution of each pixel in the original image to the target class.

- **Integrated Gradients**: Estimates pixel importance by integrating gradients along a path from a baseline image to the actual image. This technique can be memory-intensive, so specific parameters must be set to avoid memory saturation. In particular, $n\_steps = 200$ represents the number of interpolation steps, and $internal\_batch\_size = 10$ indicates that the algorithm divides the total number of steps $\times$ examples into chunks of size at most 10, which are processed sequentially.

- **Kernel SHAP**: A technique based on the Shapley value approach that estimates the contribution of image regions (superpixels) to the output. The first design choice for this method is how to generate the feature mask; in this case, SLIC [22] was used with 100 superpixels. Two parameters were tuned for this method: $n\_samples = 400$, representing the number of samples of the original model used to train the surrogate interpretable model; and $perturbations\_per\_eval = 32$, which groups several perturbations together in a single forward pass (as a batch), speeding up the process.

The first three methods were also implemented with **SmoothGrad**, a technique that reduces noise by generating N noisy versions of the same input by adding Gaussian noise and then averaging them. Captum enables this strategy through the `NoiseTunnel` class. The following parameters were used:

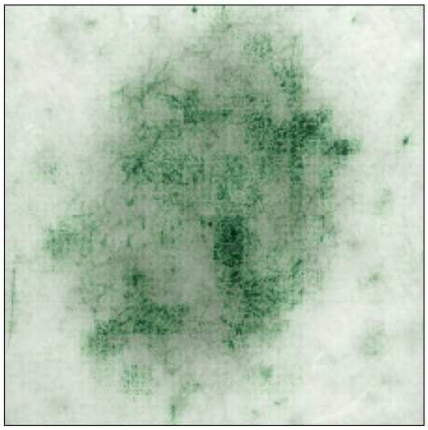
- $nt\_samples = 30$: The number of noisy versions of the input image generated.
- $nt\_samples\_batch\_size = 5$: The batch size used to process the noisy samples.
- $stdev = 0.2$: The standard deviation of the Gaussian noise added to the input image.

To provide a qualitative comparison of the different explanation techniques, Figure 3.3 reports an illustrative example generated from the same dermatological image. The figure shows the resulting saliency maps obtained using the five methods under evaluation. This visualization highlights the distinct behavior of each technique in terms of localization, precision, and comprehensibility of the explanations.
For each technique, the continuous attribution values were converted into binary masks. This was achieved through normalization followed by percentile thresholding: only pixels above the 70th percentile were set to 1. The dataset used for these experiments was HAM10000 [39], which includes annotations for both classification and segmentation tasks.
The generated masks were compared with the ground-truth segmentation masks using the Intersection over Union (IoU) metric. The IoU is defined as the ratio between the intersection and the union of the predicted mask P and the ground-truth mask G:

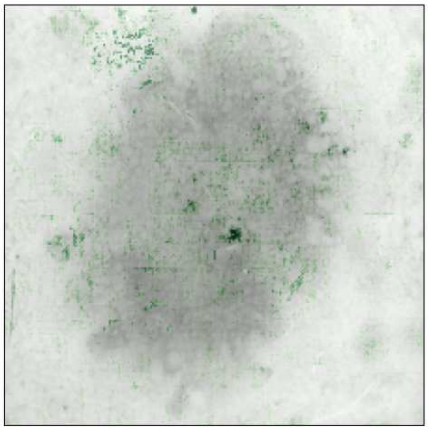
$$\text{IoU}(P, G) = \frac{|P \cap G|}{|P \cup G|}$$

When evaluating across a dataset of N images, the mean IoU (mIoU) is computed as the average IoU over all samples:

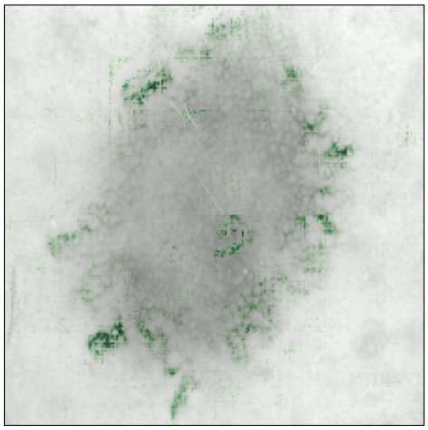
$$\text{mIoU} = \frac{1}{N} \sum_{i=1}^{N} \text{IoU}(P_i, G_i)$$



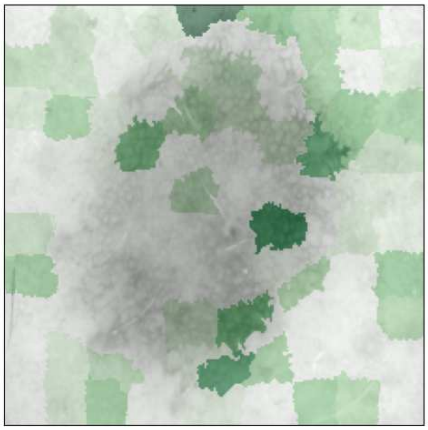
(a) Vanilla Gradient



(b) Input × Gradient



(c) Integrated Gradients



(d) Kernel SHAP

Figure 3.3: Example of saliency maps generated for the same dermatological image using different explanation techniques: (a) Vanilla Gradient, (b) Input × Gradient, (c) Integrated Gradients, and (d) Kernel SHAP.

3.4.2 Results

The evaluation was conducted on 320 images from HAM10000, equally split between correctly and incorrectly classified samples. The mean IoU results for each method are shown in Table 3.8.

Table 3.8: Mean Intersection over Union (mIoU) between the saliency maps and the ground-truth segmentation masks on 320 images from the HAM10000 dataset.

Explanation Technique	**mIoU**
Integrated Gradients	0.242
Input × Gradient	0.234
Vanilla Gradient	**0.421**
Kernel SHAP	0.191

The best-performing technique was **Vanilla Gradient**, which outperformed more complex approaches. This finding is notable, as it suggests that a simple gradient-based method can better capture clinically relevant regions compared to theoretically stronger but more complex algorithms. Kernel SHAP obtained the lowest scores, likely due to its reliance on superpixels, which do not always align with lesion boundaries. Integrated Gradients and Input × Gradient achieved comparable intermediate results.

3.4.3 Error Analysis and Central Agent Perspective

A second analysis focused on misclassified images. The goal was to investigate whether saliency maps could reveal that the model was attending to incorrect image regions, particularly when the classification was wrong.
With this objective, the IoU values were compared between Vanilla Gradient attributions and the ground-truth segmentation masks for the subset of correctly classified images. For the subset of misclassified images, the explanations were generated twice: first by targeting the predicted class, and then by targeting the ground-truth class. The results are shown in Table 3.9.

Table 3.9: Mean IoU (mIoU) values obtained when comparing saliency maps generated with Vanilla Gradient against ground-truth segmentation masks. Results are reported for correctly classified images and for misclassified images, using both the predicted class and the ground-truth class as targets.

Scenario	mIoU
Correct predictions	**0.451**
Wrong predictions, target = predicted class	0.395
Wrong predictions, target = ground-truth class	0.392

The small difference between the latter two values suggests that, in misclassified cases, saliency maps do not realign with the ground-truth class. Nevertheless, the consistent difference observed between correctly classified and misclassified images highlights a potentially valuable signal for the central reasoning agent: it could detect unreliable predictions through the explanations and trigger alternative reasoning strategies.

3.4.4 Evaluation of SmoothGrad

Finally, the impact of the SmoothGrad technique was assessed. SmoothGrad reduces noise in gradient-based explanations by averaging attributions from multiple noisy versions of the same input. Three methods were tested with and without SmoothGrad. The results are shown in Table 3.10.

Table 3.10: Comparison of mean IoU (mIoU) with and without the application of SmoothGrad (SG). Results are reported for the three gradient-based explanation techniques. SmoothGrad consistently improves the overlap with the ground-truth segmentation masks.

Explanation Technique	mIoU without SG	mIoU with SG
Integrated Gradients	0.205	0.242
Input × Gradient	0.192	0.234
Vanilla Gradient	0.301	**0.421**

In all cases, SmoothGrad consistently improved mIoU, confirming its effectiveness in generating clearer and more clinically aligned saliency maps.

Chapter 4

Conclusions

This thesis has presented the design, architecture, and evaluation of an AI agent framework for the diagnostic support of dermatologists. The system integrates a reasoning core, based on a multimodal Large Language Model, with a set of specialized tools for image classification, lesion detection, patient metadata integration, and explainability.
The central idea is to produce a chatbot that not only generates accurate results but also explains its decisions in a transparent way, incorporating the expertise of dermatologists into the reasoning loop. To achieve these results, the system combines general-purpose reasoning with domain-specific models, avoiding retraining or fine-tuning of the LLMs with medical data, and creating a framework that is scalable and customizable for every specific use.
The experimental results highlight several key findings. First, general-purpose multimodal LLMs struggle to classify dermatological images, confirming that domain knowledge is essential in medical applications. Second, the integration of vision tools such as the SwinDI classifier significantly improves performance, demonstrating the effectiveness of a modular design and highlighting the LLMs' capabilities in orchestrating tools. Third, the incorporation of patient metadata, encoded as text embeddings, provides measurable improvements, showing the effectiveness of contextual information in the clinical workflow and the robustness of the chosen integration strategy.
Moreover, the experiments with Concept Bottleneck Models illustrate the potential of aligning predictions with human-understandable concepts, improving interpretability. However, the limited amount of dermatological images annotated at the concept level constrains the effectiveness of this technique, revealing a trade-off that is more unbalanced toward model transparency than performance.

Looking ahead, several directions for future work can be identified. A possible enhancement of the Concept Bottleneck Model could be the adoption of Concept Embedding Models [46]. Instead of explicitly predicting concepts before

the final diagnosis, concept embeddings would represent them in a continuous high-dimensional latent space, capturing richer relationships and providing better generalization. These embeddings can also be aligned with textual or medical knowledge representations and do not strictly require every training image to be annotated with concepts. In fact, CEMs can leverage large unannotated datasets by aligning to embedding spaces. Moreover, this technique allows the incorporation of new concepts without retraining the models, as they can naturally be embedded into the same representation space.

Further research could also investigate attention-based explanation methods. In this thesis, gradient-based and perturbation-based saliency methods were employed, but since the model to explain is based on a Transformer, attention can also be used to retrieve the attributions for the saliency map. In the literature, methods have been proposed that leverage attention to produce faithful and effective explanations, showing that this direction can provide high-quality attribution maps [47]. Applying these strategies to the Swin Transformer remains challenging due to its hierarchical structure, which does not produce fixed-size attention maps. Leveraging attention mechanisms directly for interpretability could yield explanations more faithful to the model's reasoning.

Finally, an exciting perspective could be the development of multi-agent systems. In this context, different agents could cooperate by specializing in subtasks or interact in an adversarial way to challenge each other's predictions, ultimately improving robustness. These multi-agent dynamics could reflect the collaborative and critical discussions of clinical teams, enhancing both accuracy and reliability.

Bibliography

[1] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. «Concept Bottleneck Models». In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 13–18, 2020, pp. 5338–5348. URL: https://proceedings.mlr.press/v119/koh20a.html (cit. on pp. 3, 11, 18, 19, 31).

[2] Wei Xu and Alexander Rudnicky. «Can artificial neural networks learn language models?» In: (2000) (cit. on p. 6).

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. «Efficient estimation of word representations in vector space». In: *arXiv preprint arXiv:1301.3781* (2013) (cit. on p. 6).

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. «Attention is all you need». In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 6, 8).

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. «Bert: Pre-training of deep bidirectional transformers for language understanding». In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186 (cit. on pp. 7, 8).

[6] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. «Exploring the limits of transfer learning with a unified text-to-text transformer». In: *Journal of machine learning research* 21.140 (2020), pp. 1–67 (cit. on p. 7).

[7] Ranjan Sapkota, Konstantinos I. Roumeliotis, and Manoj Karkee. «AI Agents vs. Agentic AI: A Conceptual taxonomy, applications and challenges». In: *Information Fusion* 126 (2026), p. 103599. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2025.103599. URL: https://www.sciencedirect.com/science/article/pii/S1566253525006712 (cit. on p. 8).

[8] Adibvafa Fallahpour, Jun Ma, Alif Munim, Hongwei Lyu, and Bo Wang. *MedRAX: Medical Reasoning Agent for Chest X-ray*. 2025. arXiv: 2502.02673 [cs.LG]. URL: https://arxiv.org/abs/2502.02673 (cit. on p. 8).

[9] Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B Divya. «Agentic ai: Autonomous intelligence for complex goals–a comprehensive survey». In: *IEEe Access* (2025) (cit. on p. 8).

[10] Jinghao Feng, Qiaoyu Zheng, Chaoyi Wu, Ziheng Zhao, Ya Zhang, Yanfeng Wang, and Weidi Xie. «Mˆ 3Builder: A Multi-Agent System for Automated Machine Learning in Medical Imaging». In: *arXiv preprint arXiv:2502.20301* (2025) (cit. on p. 8).

[11] Mohamed H Shahin, Srijib Goswami, Sebastian Lobentanzer, and Brian W Corrigan. «Agents for Change: Artificial Intelligent Workflows for Quantitative Clinical Pharmacology and Translational Sciences». In: *Clinical and Translational Science* 18.3 (2025), e70188 (cit. on p. 9).

[12] Loris Cino, Cosimo Distante, Alessandro Martella, and Pier Luigi Mazzeo. «Skin Lesion Classification Through Test Time Augmentation and Explainable Artificial Intelligence». In: *Journal of Imaging* 11.1 (2025), p. 15. DOI: 10.3390/jimaging11010015 (cit. on p. 10).

[13] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. «Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on p. 10).

[14] Laurens van der Maaten and Geoffrey Hinton. «Visualizing data using t-SNE». In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605 (cit. on p. 10).

[15] Bo Lin, Yingjing Xu, Xuanwen Bao, Zhou Zhao, Zhouyang Wang, and Jianwei Yin. «SkinGEN: An explainable dermatology diagnosis-to-generation framework with interactive vision-language models». In: *Proceedings of the 30th International Conference on Intelligent User Interfaces*. 2025, pp. 1287–1296 (cit. on p. 10).

[16] Erik Štrumbelj and Igor Kononenko. «Explaining prediction models and individual predictions with feature contributions». In: *Knowledge and information systems* 41.3 (2014), pp. 647–665 (cit. on p. 11).

[17] Ian Covert, Scott Lundberg, and Su-In Lee. «Explaining by removing: A unified framework for model explanation». In: *Journal of Machine Learning Research* 22.209 (2021), pp. 1–90 (cit. on p. 11).

[18] Eunji Kim, Siwon Kim, Minji Seo, and Sungroh Yoon. «Xprotonet: diagnosis in chest radiography with global and local explanations». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 15719–15728 (cit. on p. 11).

[19] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. «Gradient-based attribution methods». In: *Explainable AI: Interpreting, explaining and visualizing deep learning*. Springer, 2019, pp. 169–191 (cit. on p. 11).

[20] Harshit Bokadia, Scott Cheng-Hsin Yang, Zhaobin Li, Tomas Folke, and Patrick Shafto. «Evaluating perceptual and semantic interpretability of saliency methods: A case study of melanoma». In: *Applied AI Letters* 3.3 (2022), e77 (cit. on p. 12).

[21] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. «Axiomatic Attribution for Deep Networks». In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, June 2017, pp. 3319–3328. URL: https://proceedings.mlr.press/v70/sundararajan17a.html (cit. on p. 13).

[22] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. «SLIC Superpixels Compared to State-of-the-Art Superpixel Methods». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120 (cit. on pp. 16, 52).

[23] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. «Rich feature hierarchies for accurate object detection and semantic segmentation». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587 (cit. on p. 20).

[24] Ross Girshick. «Fast R-CNN». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015 (cit. on p. 20).

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1137–1149. DOI: 10.1109/TPAMI.2016.2577031 (cit. on p. 20).

[26] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. «Mask R-CNN». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on p. 20).

[27] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. «Dab-detr: Dynamic anchor boxes are better queries for detr». In: *arXiv preprint arXiv:2201.12329* (2022) (cit. on p. 20).

[28] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. «Dn-detr: Accelerate detr training by introducing query denoising». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 13619–13627 (cit. on p. 20).

[29] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. *DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection*. 2022. arXiv: 2203.03605 [cs.CV] (cit. on p. 20).

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. «Microsoft COCO: Common Objects in Context». In: *Computer Vision – ECCV 2014*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1 (cit. on p. 20).

[31] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. «Masked-attention mask transformer for universal image segmentation». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 1290–1299 (cit. on p. 21).

[32] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M. Ni, and Heung-Yeung Shum. «Mask DINO: Towards a Unified Transformer-Based Framework for Object Detection and Segmentation». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 3041–3050 (cit. on p. 21).

[33] Konstantinos Liopyris, Stamatios Gregoriou, Julia Dias, and Alexandros J. Stratigos. «Artificial Intelligence in Dermatology: Challenges and Perspectives». In: *Dermatology and Therapy* 12.12 (2022), pp. 2637–2651. DOI: 10.1007/s13555-022-00833-8. URL: https://doi.org/10.1007/s13555-022-00833-8 (cit. on p. 21).

[34] The International Skin Imaging Collaboration. *ISIC Archive*. https://www.isic-archive.com/. Accessed: May 20, 2024 (cit. on p. 21).

[35] Roxana Daneshjou et al. «Disparities in dermatology AI performance on a diverse, curated clinical image set». In: *Science Advances* 8.31 (2022), eabq6147. DOI: 10.1126/sciadv.abq6147 (cit. on p. 21).

[36] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. «Dermatologist-level classification of skin cancer with deep neural networks». In: *Nature* 542.7639 (2017), pp. 115–118. DOI: 10.1038/nature21056. URL: https://doi.org/10.1038/nature21056 (cit. on p. 21).

[37] Seung Seog Han, Ilwoo Park, Sung Eun Chang, Woohyung Lim, Myoung Shin Kim, Gyeong Hun Park, Je Byeong Chae, Chang Hun Huh, and Jung-Im Na. «Augmented Intelligence Dermatology: Deep Neural Networks Empower Medical Professionals in Diagnosing Skin Cancer and Predicting Treatment Options for 134 Skin Disorders». In: *Journal of Investigative Dermatology* 140.9 (2020), pp. 1753–1761. ISSN: 0022-202X. DOI: https://doi.org/10.1016/j.jid.2020.01.019. URL: https://www.sciencedirect.com/science/article/pii/S0022202X20301366 (cit. on p. 21).

[38] Mirco Gallazzi, Sara Biavaschi, Alessandro Bulgheroni, Tommaso M. Gatti, Silvia Corchs, and Ignazio Gallo. «A Large Dataset to Enhance Skin Cancer Classification With Transformer-Based Deep Neural Networks». In: *IEEE Access* 12 (2024), pp. 109544–109559. DOI: 10.1109/ACCESS.2024.3439365 (cit. on p. 22).

[39] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. «The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions». In: *Scientific data* 5.1 (2018), pp. 1–9 (cit. on pp. 30, 45, 53).

[40] Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh. «Seven-point checklist and skin lesion classification using multitask multimodal neural nets». In: *IEEE journal of biomedical and health informatics* 23.2 (2018), pp. 538–546 (cit. on p. 30).

[41] Marc Combalia et al. «Bcn20000: Dermoscopic lesions in the wild». In: *arXiv preprint arXiv:1908.02288* (2019) (cit. on pp. 30, 45).

[42] Amanda Oakley, Mark Duffill, and Marius Rademaker. *DermNet New Zealand trust*. [Online. Accessed November 12, 2024]. 1996. URL: https://dermnetnz.org/ (cit. on p. 30).

[43] Matthew Groh, Caleb Harris, Luis Soenksen, Felix Lau, Rachel Han, Aerin Kim, Arash Koochek, and Omar Badri. «Evaluating deep neural networks trained on clinical images in dermatology with the fitzpatrick 17k dataset». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 1820–1828 (cit. on p. 31).

[44] Roxana Daneshjou, Mert Yuksekgonul, Zhuo Ran Cai, Roberto Novoa, and James Y Zou. «Skincon: A skin disease dataset densely annotated by domain experts for fine-grained debugging and analysis». In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 18157–18167 (cit. on pp. 31, 48).

[45] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. «Imagenet: A large-scale hierarchical image database». In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on p. 41).

[46] Mateo Espinosa Zarlenga et al. «Concept embedding models: Beyond the accuracy-explainability trade-off». In: *Advances in neural information processing systems* 35 (2022), pp. 21400–21413 (cit. on p. 57).

[47] Hila Chefer, Shir Gur, and Lior Wolf. «Transformer interpretability beyond attention visualization». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 782–791 (cit. on p. 58).