"Cloud-Based Skin Cancer Detection Using Deep Learning and Scalable AWS Infrastructure"

Module Assignment for

CS5024 - Theory and Practice of Advanced AI Ecosystems

Student Name : Steven Gerard Mascarenhas

Student ID : 24044407

Revision Timestamp: 30/05/2025 18:22:55

Module Director: Patrick Denny
Teaching Assistant: Tejus Vignesh Vijayakumar

Abstract

Skin cancer is among the most common cancers globally, with over 1.5 million cases reported annually. Early detection is critical for successful treatment outcomes, but the manual interpretation of dermatoscopic images requires expert knowledge and is often inaccessible in low-resource regions (International Agency for Research on Cancer, 2022).Deep learning has emerged as a transformative approach in medical image analysis. Vision Transformer (ViT) models, in particular, have demonstrated high accuracy and robustness in segmenting and classifying skin lesions from dermatoscopic images, making them well-suited for non-invasive diagnostic support systems (Himel et al, 2024).

This project deploys a fine-tuned Vision Transformer (ViT) model from Hugging Face on the AWS ecosystem to classify dermatoscopic images into seven common skin lesion categories. Leveraging the AWS AI Ecosystem, the architecture includes Amazon S3 for image storage, AWS Lambda for event-driven preprocessing and inference orchestration, Amazon SageMaker for scalable model hosting, and Streamlit for user interaction.

The motivation stems from the growing prevalence of skin cancer and the demand for faster, scalable diagnostic support systems, coupled with my personal interest in applying deep learning skills to impactful healthcare use cases. The system showcases how cloud-native AI services like SageMaker, Lambda, and S3 can be integrated to create scalable, cost-efficient, and real-time diagnostic tools, aligning with the AWS Well-Architected Framework (Patrick Denny, 2025). This makes it an ideal portfolio project for healthcare-focused AI solutions that emphasize practical deployment and operational scalability.

Contents

Abstract 2

Introduction 3

AI Ecosystem Architecture Used 4

.... 4

AWS Services Used 5

Model Description 9

Scalability Considerations 10

References 12

Figures

Figure 1:Architecture diagram used to create the AI/ML Ecosystem 4
Figure 2: AWS SageMaker Notebook Instance. 5
Figure 3:AWS SageMaker Endpoint 5
Figure 4:Lambda function 6
Figure 5:Pillow layer for the lambda function 6
Figure 6:S3 trigger to invoke the lambda function 7
Figure 7:S3 Bucket with the uploads and predictions folders 6
Figure 8::SageMaker role with polices 8
Figure 9::Lambda role with policies 8
Figure 10:Cloud Watch Log Groups 9
Figure 11:Cloud Watch Billing Alarm 9
Figure 12:Full Prediction Output of the different skin condtions along with their confidence score 10
Figure 13:Streamlit interface with top prediction output of the skin condition and along with the confidence score. 10

Introduction

Skin cancer is a growing health concern, and early detection is key to effective treatment. In many areas, access to expert diagnosis is limited, so the ability to quickly assess a skin lesion through a simple image upload can be transformative.

This project uses the AWS AI Ecosystem to create a scalable and secure system for skin cancer detection. It is powered by a fine-tuned Vision Transformer (ViT) model from Hugging Face, trained to identify conditions like melanoma, basal cell carcinoma, and melanocytic nevi etc (Anwarkh1, 2024). The model runs on an Amazon SageMaker endpoint, enabling real-time predictions without manual infrastructure setup. When a user uploads an image, it is saved in Amazon S3, which triggers an AWS Lambda function to preprocess the image and send it to SageMaker.The result is returned in JSON format, saved back to S3, and shown to the user.

This setup uses serverless and managed services to keep things efficient and cost-effective. IAM roles handle secure access between services, CloudWatch monitors performance and logs, and resources like Lambda and ml.t2.medium instances help stay within the AWS free tier. By following AWS best practices, the system remains reliable, scalable, and affordable, showing how cloud-based AI can support real-world healthcare needs (Patrick Denny, 2025).

AI Ecosystem Architecture Used

The architecture of this project is built around a user-friendly Streamlit frontend, which allows users to upload dermatoscopic images for classification. Once an image is uploaded, it is stored in an Amazon S3 bucket (skin-cancer-demo-bucket) within the uploads/ folder. This upload action triggers an AWS Lambda function via S3 event notification (Be A Better Dev, 2022).

The Lambda function acts as the core preprocessing and orchestration layer. It retrieves the uploaded image, converts it to RGB, resizes it to 224×224 pixels, and re-encodes it as a JPEG to meet the input requirements of the pretrained Vision Transformer (ViT) model. Although preprocessing is handled in the Lambda function during production, similar steps are included in the SageMaker notebook used during development to ensure consistency and aid in local testing and debugging.

After preprocessing, the Lambda function sends the image as raw JPEG bytes to an Amazon SageMaker inference endpoint. This endpoint hosts a fine-tuned ViT model sourced from Hugging Face (Anwarkh1, 2024), trained to classify images into seven categories of skin lesions. The endpoint processes the image and returns predictions in JSON format, including labels and their associated confidence scores.

These predictions are then written back to the predictions/ folder of the same S3 bucket. The results are subsequently retrieved and displayed to the user through the Streamlit interface, closing the loop from image upload to diagnosis output.

This end-to-end pipeline demonstrates how modern deep learning models can be deployed at scale in real-world medical applications using AWS cloud infrastructure, aligning with the principles outlined in the AWS Well-Architected Framework (Patrick Denny, 2025,)

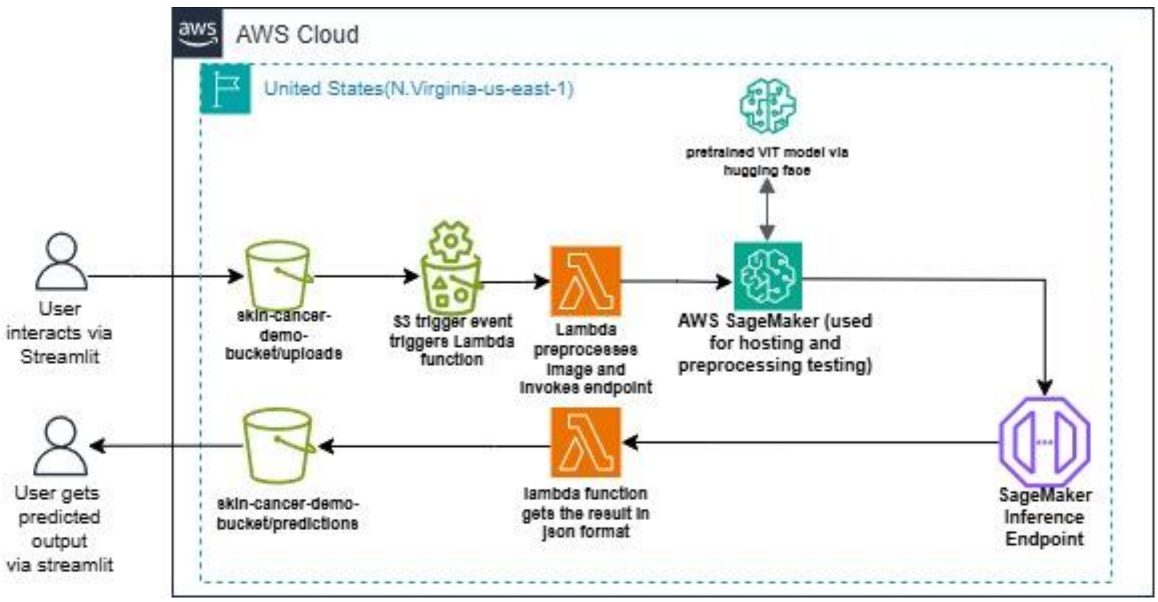

Figure 1:Architecture diagram used to create the AI/ML Ecosystem

AWS Services Used

1.Amazon Sagemaker

I used the Amazon SageMaker Python SDK (Amazon Web Services, 2024) to deploy a fine-tuned Vision Transformer (ViT) model from Hugging Face for real-time skin cancer classification. The SDK simplified the deployment process by abstracting infrastructure complexities and handling AWS resource integrations (Patrick Denny, 2025). I launched a SageMaker notebook instance named skin-cancer-notebook1 using an ml.t3.medium EC2 instance (free-tier eligible) as shown in Figure 2, which was sufficient for hosting the model and performing inference at scale (Patrick Denny, 2025).

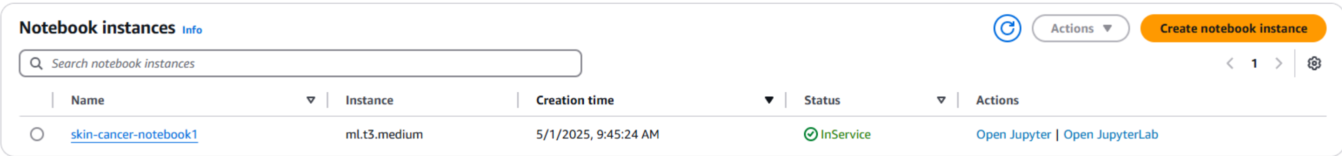

Figure 2: AWS SageMaker Notebook Instance.

The model was successfully deployed to an endpoint named huggingface-pytorch-inference-2025-05-02-11-28-45-405, as shown in Figure 3.

Figure 3:AWS SageMaker Endpoint

The deployed model receives preprocessed JPEG images and returns predictions as a JSON list of labels with confidence scores. These are parsed by a Lambda function, stored in S3, and displayed on the Streamlit frontend . SageMaker simplified deployment while enabling scalable, real-time inference with minimal infrastructure management.

2.AWS Lambda

AWS Lambda acts as the serverless backbone of the inference workflow in this architecture. When a new image is uploaded to the uploads/ folder in the Amazon S3 bucket, it triggers the skin-cancer-predictor-fn Lambda function Figure 4. This function is responsible for downloading the image from S3, preprocessing it by converting it to RGB and resizing it to 224×224 pixels using the Pillow library (enabled through a Lambda layer, as seen in Figure 5), and then sending the raw image bytes to the SageMaker endpoint specified through an environment variable.The endpoint returns predictions in JSON format, which the function then saves in the predictions/ folder of the same S3 bucket.

The Lambda function operates under the lambda_execution_role3, which includes policies such as AmazonS3FullAccess, AmazonSageMakerFullAccess, and AWSLambdaBasicExecutionRole to ensure secure and authorized

interaction with S3, SageMaker, and CloudWatch. This event-driven approach ensures low-latency, scalable processing and adheres to the AWS Well-Architected Framework for serverless applications (Patrick Denny, 2025).

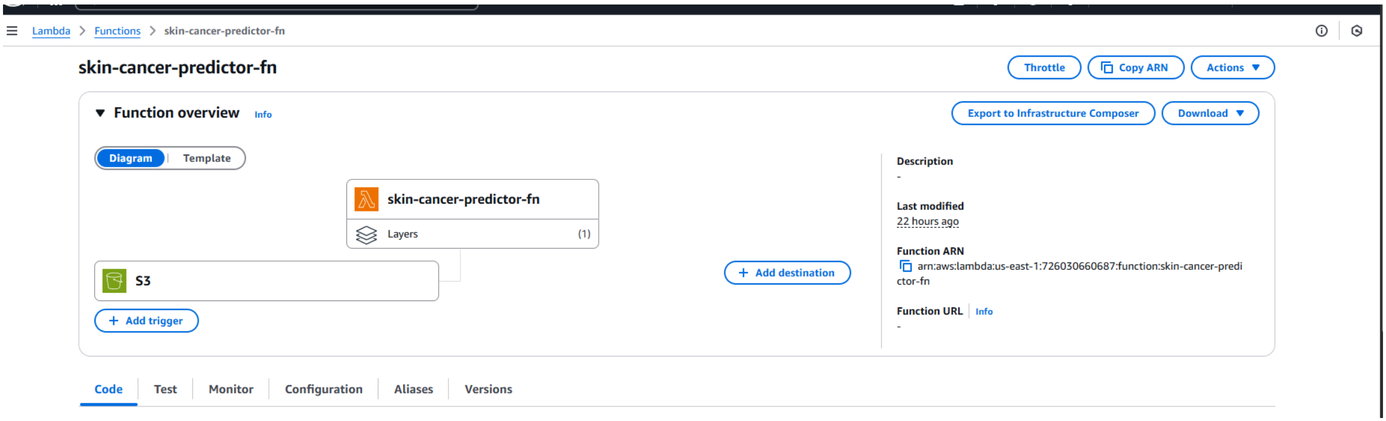

Figure 4:Lambda function

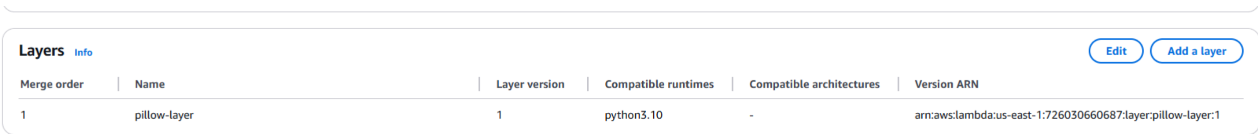

Figure 5:Pillow layer for the lambda function

3.AWS S3

Amazon Simple Storage Service (S3) serves as the central data hub in this project, enabling scalable, durable, and highly available storage for skin lesion images and their corresponding prediction outputs. A single bucket named skin-cancer-demo-bucket as shown is Figure 7 is structured with two logical folders: uploads/ for user-submitted images and predictions/ for storing the inference results returned from SageMaker. When a user uploads an image through the Streamlit web interface, it is stored in the uploads/ folder of this bucket (Patrick Denny, 2025).

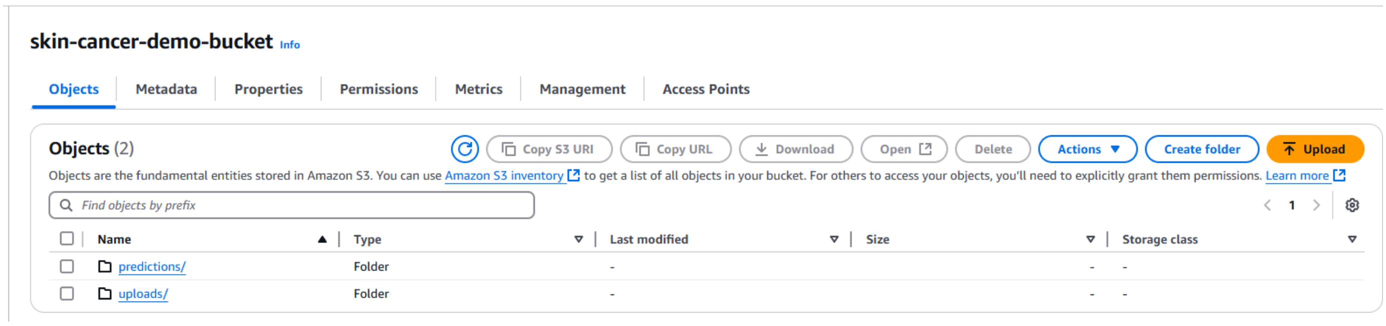

Figure 6:S3 Bucket with the uploads and predictions folders

The S3 bucket is configured with an event notification that triggers an AWS Lambda function upon the creation of a new object in the uploads/ folder as shown in Figure 6. This setup allows seamless automation of the inference process without requiring continuous polling. Once the Lambda function is triggered, it downloads the uploaded image, preprocesses it, invokes the SageMaker inference endpoint, and stores the resulting predictions back into the predictions/ folder in JSON format (Amazon Web Services, 2020). This design pattern aligns with AWS best practices by leveraging event-driven architecture for serverless orchestration, ensuring low latency and high scalability. Additionally, access to the S3 bucket is controlled using IAM roles and policies attached to Lambda and SageMaker, ensuring secure and restricted access to only authorized services.

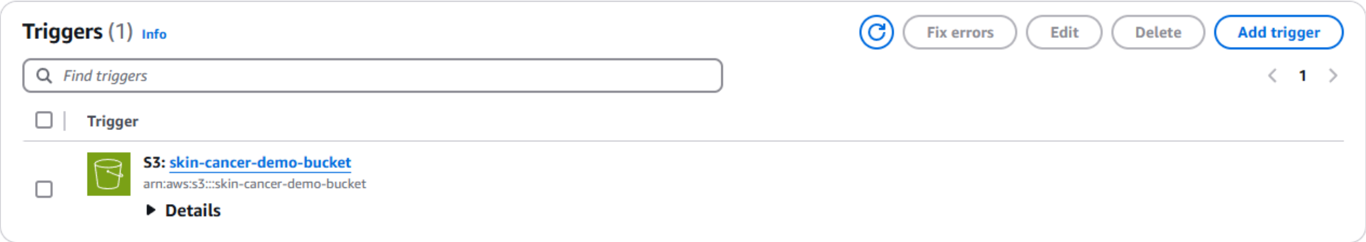

Figure 7:S3 trigger to invoke the lambda function

4.AWS IAM Roles

AWS Identity and Access Management (IAM) played a vital role in securing access across the various AWS services used in my skin cancer classification pipeline. By defining explicit roles and policies, IAM ensured that each service (SageMaker, Lambda, S3) only had the minimum permissions required to perform its tasks. I configured two key roles such as sagemaker_execution_role3 as shown in Figure 8 for my SageMaker notebook instance, which included policies like AmazonS3FullAccess and AmazonSageMakerFullAccess, enabling model deployment and S3 interaction. Similarly,The Lambda role included AmazonS3FullAccess, AmazonSageMakerFullAccess, and AWSLambdaBasicExecutionRole as shown in Figure 9 allowing it to fetch images from S3, invoke the SageMaker endpoint, save predictions, and log to CloudWatch. These roles also supported programmatic access using the Boto3 SDK, ensuring that service interactions such as S3 file uploads or inference requests were securely handled under the principle of least privilege (Patrick Denny, 2025).

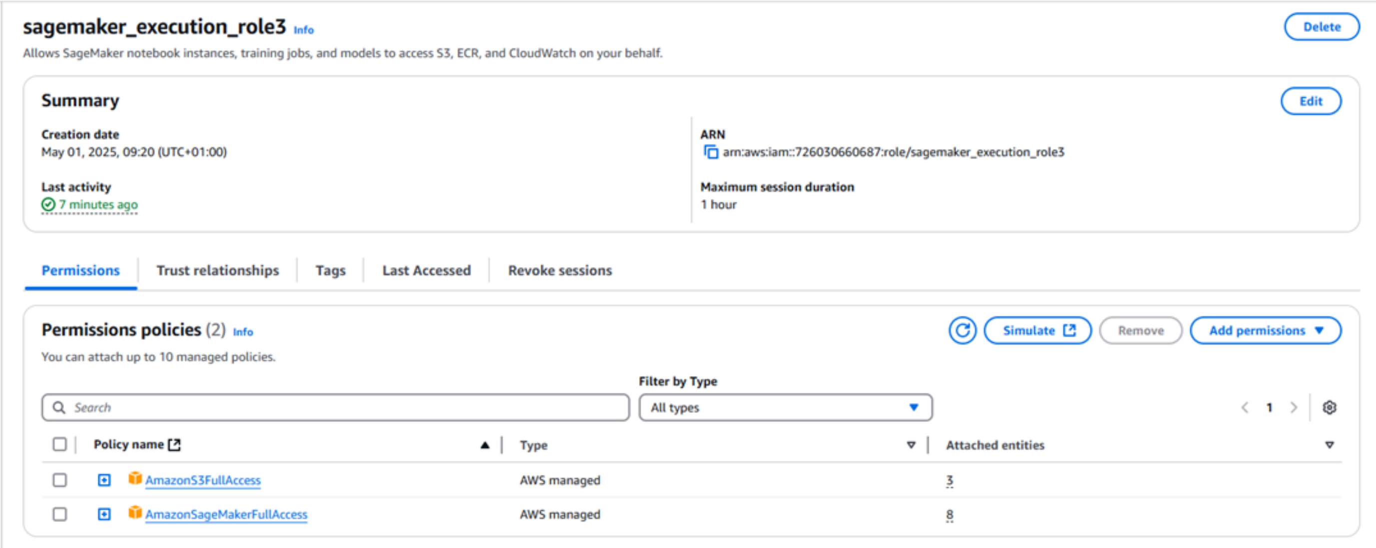

Figure 8::SageMaker role with polices

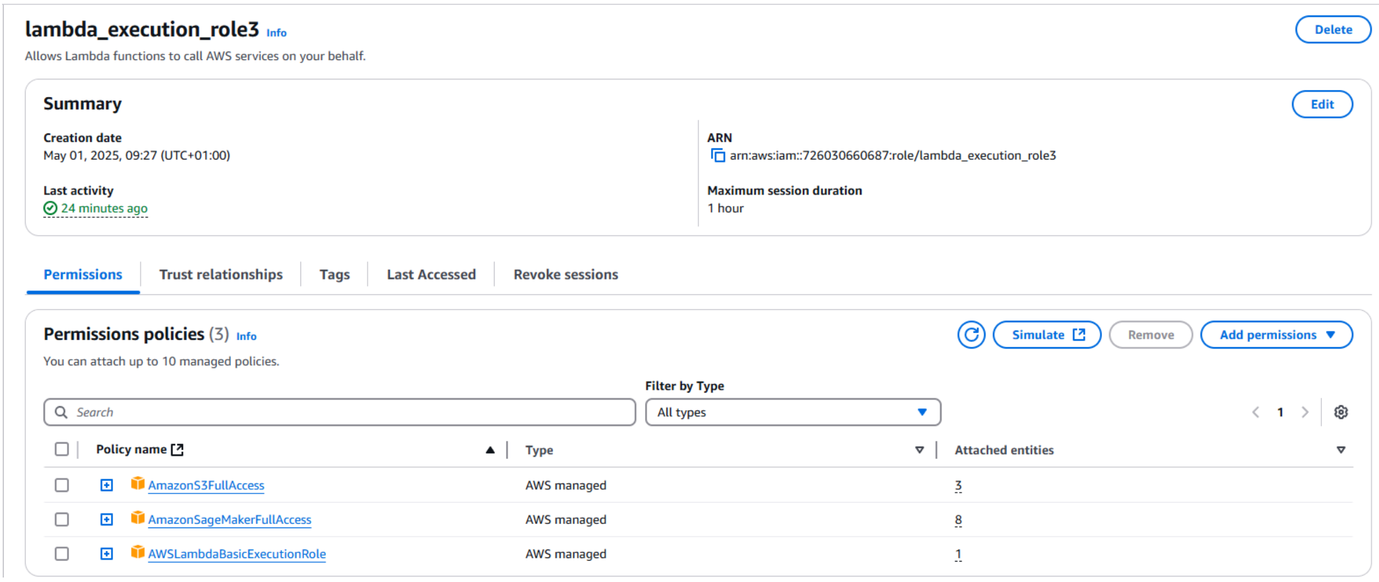

Figure 9::Lambda role with policies

5.Cloud Watch

AWS CloudWatch was used to monitor and log activities across three essential components such as the Lambda function, the SageMaker endpoint, and the SageMaker notebook instance as shown in Figure 10. The log group /aws/lambda/skin-cancer-predictor-fn recorded all Lambda executions, including image preprocessing, inference requests, and responses, making it crucial for debugging and tracking S3-triggered events. The log group /aws/sagemaker/Endpoints/huggingface-pytorch-inference-2025-05-02-11-28-45-405 captured interactions with the deployed inference endpoint, helping identify any model-side issues.Meanwhile, /aws/sagemaker/NotebookInstances logged development activity and testing carried out in the notebook instance (Patrick Denny, 2025).

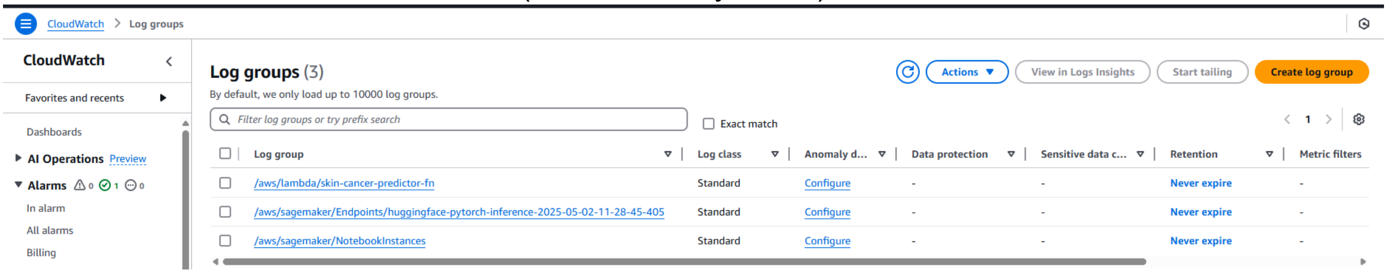

Figure 10:Cloud Watch Log Groups

Additionally, I configured a billing alarm to alert me if AWS usage exceeded free-tier limits as shown in Figure 11

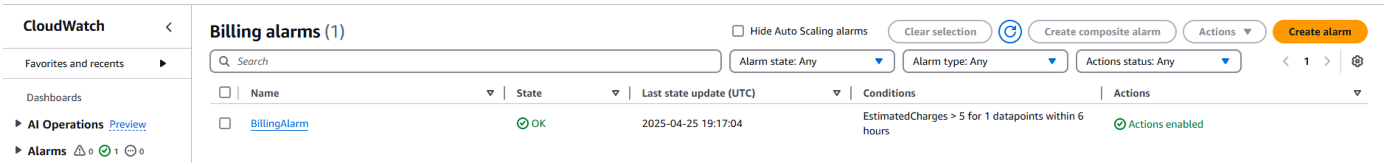

Figure 11:Cloud Watch Billing Alarm

Model Description

This project uses a fine-tuned Vision Transformer (ViT) model for skin cancer classification from hugging face, deployed using Amazon SageMaker SDK (Amazon Web Services, 2024) on SageMaker and triggered via AWS Lambda through an S3 event-driven pipeline (Be A Better Dev, 2022). The ViT architecture, originally introduced by (Alexey Dosovitskiy et al, 2021),replaces traditional convolutional layers with a transformer-based encoder, allowing it to effectively process image patches for classification. The base model used here was pre-trained on the ImageNet21k dataset and modified with a new classification head to suit the skin cancer domain (Anwarkh1, 2024).

The dataset used for training is Marmal88's Skin Cancer Dataset (Marmal88's, 2023) on Hugging Face, derived from the HAM10000 dataset, and contains seven diagnostic categories: actinic keratoses, basal cell carcinoma, benign keratosis-like lesions, dermatofibroma, melanoma, melanocytic nevi, and vascular lesions (Tschandl, Philipp, 2018). The training was performed over 5 epochs using the Adam optimizer with a learning rate of 1e-4 and a batch size of 32. Validation accuracy improved from 83.55% to 96.95%, reflecting robust generalization (Anwarkh1, 2024).

The model was deployed as a Hugging Face model in SageMaker using the HuggingFaceModel class. A Lambda function handles image preprocessing (RGB conversion, resizing to 224×224 pixels, and encoding to JPEG), sends the image to the SageMaker endpoint for inference, and stores the returned JSON predictions back to S3.

As shown in Figure 13,The image was classified the image as “actinic keratoses” with 99.1% confidence, validating the real-time capability and accuracy of the deployed system. Figure 12 illustrates the Full output predictions of the different skin conditions Streamlit interface used for inference.

This implementation demonstrates how pre-trained ViT models and AWS services can together deliver scalable, accurate, and production-ready solutions in medical image classification.

.

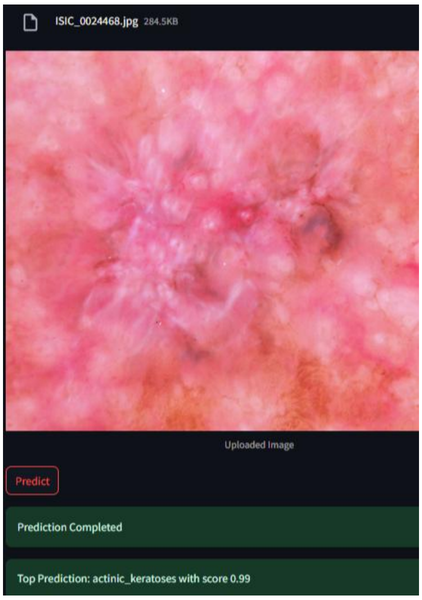

Figure 13:Streamlit interface with top prediction output of the skin condition and along with the confidence score.

Figure 12:Full Prediction Output of the different skin condtions along with their confidence score

Scalability Considerations

The architecture of this skin cancer image classification system is designed with scalability in mind, leveraging AWS managed and serverless services to ensure the solution can handle increasing user traffic, growing image datasets, and rising inference demand without requiring architectural overhaul.

Event-Driven Compute Scaling:
AWS Lambda enables automatic horizontal scaling. When multiple users upload images in parallel, each upload event triggers its own Lambda instance (Amazon Web Services, 2024). This elastic behavior allows the system to respond to spikes in usage without performance degradation, as compute resources are allocated in real time (Patrick Denny, 2025).In high-load situations, introducing Amazon SQS between S3 and Lambda could provide decoupling and resilience by queuing requests. (Amazon Web Services, 2024)

Storage and Throughput:
Amazon S3, as the central data store, inherently scales to accommodate millions of image uploads and predictions. It provides low-latency, high-durability storage with no management overhead. This makes it ideal for growing datasets and read/write throughput, aligning with best practices in scalable data architecture (Patrick Denny, 2025).

Database and Analytics Scalability (Optional):
If the system integrates Amazon DynamoDB for storing metadata or prediction logs in the future, it can benefit from on-demand scaling (Amazon Web Services, 2024). For advanced analytics, Amazon QuickSight (Amazon Web Services, 2024) and Athena allow querying and visualization at scale, handling larger data volumes without manual provisioning (Amazon Web Services, 2024).

Monitoring and Cost Control:
AWS CloudWatch is integrated for logging and monitoring, providing visibility into Lambda executions and endpoint health. AWS Budgets and billing alerts can be configured to monitor cloud usage, ensuring that resource scaling remains within budget constraints particularly important in the AWS Free Tier (Patrick Denny, 2025).

Security at Scale:
As the system scales, maintaining secure access control becomes critical. IAM roles already enforce least-privilege access. For enterprise expansion, access can be further refined using role hierarchies and policy versioning (Patrick Denny, 2025).Integration into a Virtual Private Cloud (VPC) could offer additional isolation and security for compliance-sensitive deployments (Patrick Denny, 2025).

References

Alexey Dosovitskiy et al. (2021, June). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. Retrieved from https://arxiv.org/abs/2010.11929

Amazon Web Services. (2020, November). *Call an Amazon SageMaker model endpoint using AWS Lambda*. Retrieved from AWS Machine Learning Blog: https://aws.amazon.com/blogs/machine-learning/call-an-amazon-sagemaker-model-endpoint-using-amazon-api-gateway-and-aws-lambda/

Amazon Web Services. (2024). *Amazon Athena Features – Serverless Interactive Query Service.* . Retrieved from Retrieved from Amazon Web Services: https://aws.amazon.com/athena/features/

Amazon Web Services. (2024). *Amazon QuickSight User Guide. Retrieved from AWS Docs:* . Retrieved from https://docs.aws.amazon.com/quicksight/latest/user/welcome.html

Amazon Web Services. (2024). *DynamoDB on-demand capacity mode. Retrieved from Amazon Web Services*. Retrieved from https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/ondemand-capacity-mode.html

Amazon Web Services. (2024). *Understanding Lambda function scaling. Retrieved from Amazon Web*. Retrieved from https://docs.aws.amazon.com/lambda/latest/dg/lambda-concurrency.html

Amazon Web Services. (2024). *Use Built-in Algorithms with Pre-trained Models in SageMaker Python SDK*. Retrieved from SageMaker Documentation: https://sagemaker.readthedocs.io/en/stable/overview.html#use-built-in-algorithms-with-pre-trained-models-in-sagemaker-python-sdk

Amazon Web Services. (2024). *What is Amazon Simple Queue Service? Retrieved from Amazon Web*. Retrieved from Retrieved from Amazon Web: https://aws.amazon.com/sqs/

Anwarkh1. (2024). *Hugging Face*. Retrieved from Skin Cancer Image Classification Model using fine-tuned Vision Transformer (ViT): https://huggingface.co/Anwarkh1/Skin_Cancer-Image_Classification

Be A Better Dev. (2022, May 24). *AWS S3 File Upload + Lambda Trigger - Step by Step Tutorial*. Retrieved from YouTube: https://www.youtube.com/watch?v=OJrxbr9ebDE&ab_channel=BeABetterDev

Himel et al. (2024). Skin Cancer Segmentation and Classification Using Vision Transformer for Automatic Analysis in Dermatoscopy-based Non-invasive Digital System. *arXiv*. Retrieved from https://arxiv.org/abs/2401.04746

International Agency for Research on Cancer. (2022). *Skin Cancer*. Retrieved from IARC: https://www.iarc.who.int/cancer-type/skin-cancer/

Marmal88's. (2023). *Marmal88's Skin Cancer Dataset on Hugging Face*. Retrieved from https://huggingface.co/datasets/marmal88/skin_cancer

Patrick Denny. (2025, February 26). *Module 4: AWS Cloud Security - Tuesday Week 4 Material.* Retrieved from Retrieved from CS5024 - Theory and Practice of Advanced AI Ecosystems: : https://learn.ul.ie/d2l/le/lessons/49289/units/925162

Patrick Denny. (2025, April). *ML3: Implementing a Machine Learning - Week 9 Material . Retrieved* . Retrieved from from CS5024 - Theory and Practice of Advanced AI Ecosystems: : https://learn.ul.ie/d2l/le/lessons/49289/units/942927

Patrick Denny. (2025, March). *Module 10: Automatic Scaling and Monitoring - Week 8. Retrieved from* . Retrieved from CS5024 - Theory and Practice of Advanced AI Ecosystems: : https://learn.ul.ie/d2l/le/lessons/49289/units/939567

Patrick Denny. (2025, January -). *Module 2: Cloud Economics and Billing - Week 3 Material. Retrieved*. (-, Editor, -, Producer, & -) Retrieved from from CS5024 - Theory and Practice of Advanced AI Ecosystems:: https://learn.ul.ie/d2l/le/lessons/49289/units/920425

Patrick Denny. (2025, February). *Module 6: Compute - Week 5 Material. Retrieved from CS5024* -. Retrieved from Theory and Practice of Advanced AI Ecosystems:: https://learn.ul.ie/d2l/le/lessons/49289/units/935307

Patrick Denny. (2025, March). *Module 7: Storage - Week 6 Material Retrieved from CS5024* -. Retrieved from Theory and Practice of Advanced AI Ecosystems: : https://learn.ul.ie/d2l/le/lessons/49289/units/935307

Patrick Denny. (2025, March 18). *Module 9: Cloud Architecture - Tuesday Week 7 Material.* Retrieved from CS5024 - Theory and Practice of Advanced AI Ecosystems: https://learn.ul.ie/d2l/le/lessons/49289/units/939207

Patrick Denny. (2025,, February 18). *Module 3: AWS Global Infrastructure Overview - Week 4 Material.* Retrieved from Retrieved from CS5024 - Theory and Practice of Advanced AI Ecosystems: : https://learn.ul.ie/d2l/le/lessons/49289/units/925162

Tschandl, Philipp. (2018). *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*. Retrieved from https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/DBW86T