Dat158 – Project
Rebecca Sahlem, Øystein Risbakk
15.11.24

Mole Scanner

Skin Cancer Detection webApp

Contents

Scope

Goal and Business Impact

Skin cancer is one of the most common types of cancer globally, and many people at risk need regular checkups. Unfortunately, long waiting times, financial challenges, or the need to track moles between appointments can make this difficult.

A trained CNN model with good accuracy could be a useful middle solution in these situations. It could help individuals monitor their moles more easily, reduce the workload for dermatologists, and make life a bit simpler while waiting for professional advice.

That being said, it's important to remember that no model is 100% accurate. A dermatologist should always have the final say, but this tool could act as a helpful support for early detection and keeping track of changes over time.

Existing Solutions and Current Problem Solving

The solution will be used as a supportive tool for individuals to monitor their moles between checkups, helping identify potential concerns early and reducing unnecessary dermatologist visits.

While similar apps like SkinVision exist, many use proprietary algorithms, raising reliability concerns.

Currently, the problem is solved through dermatologist examinations using tools like dermatoscopes, but these methods often involve long wait times and manual tracking of mole changes. Without machine learning, individuals would have to

rely on taking photos and comparing changes themselves, which is time-consuming and error-prone.

Performance Metrics and Pipeline

Business Metrics

For the project to succeed, the model needs at least 80% accuracy on test data to reliably classify moles as malignant or benign. Key metrics include accuracy, validation accuracy, and binary cross-entropy loss, which help track the model's performance during training and testing. Other metrics, like validation loss, ensure the model isn't overfitting. Latency and throughput are also important to make predictions fast and practical for real-world use. These metrics align with the goal of creating an accurate and efficient tool for monitoring moles.

Pipeline

The performance will be measured by how well the solution helps users identify suspicious moles early while reducing unnecessary visits to dermatologists. Key metrics include:

- **False negatives**: Making sure suspicious moles are flagged for medical attention.
- **False positives**: Minimizing unnecessary alarms that could lead to overreactions.
- **User feedback**: Tracking how many people find the tool reliable and helpful.

- **Dermatologist workload**: Checking if the solution helps reduce non-urgent appointments.

While not currently part of a more extensive pipeline, the model could be integrated into a more complex system in the future. The potential components include:

- **User Input System**: A mobile app or web interface where users upload images of their moles and receive predictions.
- **Preprocessing Module**: Ensures uploaded images are resized, normalized, and verified for quality (e.g., ensuring the image is clear and focuses on the mole).
- **Prediction Engine** (Current Model): The CNN processes the preprocessed images to classify moles as benign or malignant.
- **Result Delivery**: Provides predictions with confidence scores and suggestions for further action, such as seeing a dermatologist if the mole is flagged as suspicious.
- **Feedback Loop**: A feature where users or dermatologists confirm whether the prediction was accurate, improving the model in future iterations.
- **Integration with Healthcare Systems (Future)**: The app could connect with healthcare databases or dermatologist networks for streamlined referrals or consultations.

Stakeholders, Timeline, and Resources

Stakeholders

- **Individuals at risk of skin cancer**: People who need a tool to monitor their moles between dermatologist visits.
- **Dermatologists**: As this tool can help reduce workload by providing a preliminary filter for potential concerns.
- **Developers/Researchers**: Responsible for creating and improving the model, ensuring it performs accurately and reliably.
- **Healthcare organizations**: They could potentially adopt the model as part of a larger system to improve patient care.

Timeline

The project was completed over a week by two developers, focusing on dataset exploration, model building, training, cross-validation, and final testing.

Resources

The project relied on:

- Computational resources: **Kaggle's GPU-accelerated runtime** for efficient model training.
- Data resources: The **Skin Cancer: Malignant vs. Benign dataset** from Kaggle.
- Reference materials: TensorFlow documentation, curriculum books, and example codes for model improvement and deployment.

Metrics

The project will be considered a success if the model achieves at least 80% accuracy, as this would make it reliable enough to help users monitor moles between checkups. The main focus will be on accuracy and validation accuracy to check how well the model generalizes. Loss, specifically binary cross-entropy, will also be tracked to monitor training performance.
Metrics like latency and throughput are important to ensure the system runs smoothly, with quick predictions and the ability to handle multiple inputs if needed. These metrics are directly tied to the goal of creating a reliable and easy-to-use tool for early detection and monitoring of suspicious moles.

Data

For this project, we used a dataset from Kaggle called **Skin Cancer: Malignant vs. Benign**. The dataset contains labeled images of skin moles, categorized as either malignant (1) or benign (0). It provides enough data to train and evaluate the model effectively, but additional data could improve performance further. Privacy concerns are minimal as the dataset does not include personal information, but ethical considerations must still be addressed, especially when deploying the model. Users must be informed that it is not a replacement for professional medical advice.

The images are resized to 128x128 pixels and normalized to a 0-1 range for consistency and efficient processing. Minimal cleaning was needed due to the dataset's quality, but **data augmentation** was applied to enhance the model's

ability to generalize. This ensures better performance and reliability when dealing with new, unseen data.

Modeling

For this project, I chose to use a **Convolutional Neural Network (CNN)** because it's great for image classification tasks. At first, I considered using simpler models like logistic regression or a basic feedforward neural network to establish a baseline, but these wouldn't be effective for image data since they can't capture spatial features.

To estimate baseline performance, I looked at results from similar projects using the same Kaggle dataset, which showed varying accuracy depending on the complexity of the model. For improvement, I used **data augmentation** to make the model generalize better and **early stopping** to avoid overfitting.

After training, I planned to check misclassified examples to see if there were specific patterns, like confusing benign moles with malignant ones. While CNNs aren't as interpretable, I considered using visualization tools like **Grad-CAM** to understand what parts of the image the model focuses on.

Deployment

The model will be deployed as part of a simple **web application** where users can upload images of their moles to receive predictions on whether the mole is benign or malignant.

The predictions will serve as an early warning system to help users identify potential concerns before visiting a dermatologist.

Since the data is static, regular updates to the model are not necessary.

Monitoring the system will involve ensuring the webApp runs smoothly and checking for any unusual behavior in the predictions.

References

- Kaggle Dataset: **Skin Cancer - Malignant vs. Benign**
- Book: **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow**
- TensorFlow Documentation
- Kaggle Community Notebooks
- Grad-CAM Documentation
- Gradio and Hugging Face Documentation

Visualization of Learning Curves

The CNN model in my project shows promising results, with training and validation accuracy both reaching around 80%. The training loss decreases steadily, while the validation loss fluctuates a bit, which might mean the model could still improve when it comes to generalizing to unseen data.