ChatGPT

Skin Cancer Detection Using Deep Learning on the SLICE-3D Dataset

Authors: Yamini Mandadi, Nikhil Dhiman, Akash Saraf, Jaydeep D. Gondaliya, Saurabh Wankhade, Sudipta Bhatta
Affiliation: Department of Computer Science, California State University, Los Angeles

Speaker Notes:
Hello everyone, thank you for joining. Today we present our project **"Skin Cancer Detection Using Deep Learning on the SLICE-3D Dataset."** This work was done by our team as part of an academic collaboration at CSULA. In this talk, we'll cover the motivation behind using deep learning for melanoma detection, details of the new SLICE-3D dataset, our modeling approach (including how we combined images with patient metadata), the deep neural network architectures we explored, and the results we achieved. We'll conclude with insights and future directions for deploying such AI models in real-world skin cancer screening.

Slide 2: Introduction to the Problem

- Melanoma is the most lethal form of skin cancer, accounting for a majority of skin cancer deaths – yet if detected at an early stage, it's highly curable (5-year survival > 90% for Stage I) [1].
- Limited access to dermatologists in many regions leads to delayed or missed screenings; rural patients often lack specialist care, resulting in melanomas found at later, more dangerous stages [2].
- **Motivation:** Develop an AI-based screening tool for melanoma that uses standard digital images. Existing AI diagnostic tools often rely on dermoscope-quality images and controlled conditions, which limits their utility in real-world settings [3]. This project aims to bridge that gap by leveraging a novel, real-world image dataset.

Speaker Notes:
Melanoma poses a serious health risk – it's the deadliest skin cancer, but early detection makes a huge difference in outcomes [1]. The challenge is that not everyone has easy access to dermatologists or specialized imaging. In fact, dermatologist shortages in rural or underserved areas mean many people don't get timely skin checks, leading to melanomas being caught only when they've progressed [2]. This underscores our motivation: we want to bring effective melanoma screening to everyone via AI. The idea is to use deep learning on ordinary skin photos to flag suspicious lesions. Many current AI systems need high-quality dermoscopic images taken by experts, which isn't practical for broad populations [3]. Our work addresses this by using a dataset of more "real-world" skin images and developing models that can handle the variability of everyday photography.

Slide 3: Dataset Overview (SLICE-3D)

- **SLICE-3D Dataset:** ~401,000 de-identified lesion images (15 mm skin lesion crops) from 1,000+ patients, collected via 3D total-body photography at 7 dermatology centers (North America, EU, Australia, 2015–2024) [4].
- Images have smartphone-level resolution (comparable to standard photos) and include both cross-polarized and white-light captures. Each image is accompanied by rich metadata (e.g. patient age, sex, lesion body location, imaging type, lesion measurements) to mirror real clinical context [5] [6].
- Ground truth labels: lesions classified as **malignant** (melanoma or other skin cancer confirmed by pathology) or **benign** (no biopsy indication). Data was carefully cleaned to remove duplicates, non-lesions, and personal identifiers, yielding a high-quality, real-world dataset for training AI [7].

Speaker Notes:
Let's talk about our dataset, **SLICE-3D**. It stands for "Skin Lesion Image Crops from 3D photography." This dataset is extensive – over 400 thousand lesion images taken from total-body skin photographs [4]. These come from around 1,000 patients across multiple clinics worldwide, spanning 2015 to 2024, so it's very diverse. Each image is a close-up (~15 mm) of a skin lesion with resolution similar to what a smartphone camera would capture. We have both **white-light** images and **cross-polarized** images (a technique often used in dermatology) – so it covers different imaging modalities [5]. Crucially, SLICE-3D provides **metadata for each lesion**: patient demographics (age, sex), where on the body the lesion is, what type of imaging was used, plus computer-extracted features like lesion size, color contrast, border irregularity, etc. [6]. Each lesion is labeled either malignant (like melanoma or carcinoma confirmed by a pathology report) or likely benign (if it wasn't biopsied, implying no clinical suspicion) [7]. The dataset was rigorously curated: any duplicate images, images without real lesions, or any identifying info were removed. This gives us a very realistic but clean dataset, ideal for training our models to work on real-world images (not just perfect dermoscope images).

Slide 4: Sample Images from Dataset [PLACEHOLDER]

- Lesion images in SLICE-3D exhibit natural variability in appearance and imaging conditions. We see both **white-light** and **cross-polarized** photographs, with differences in lighting and background skin context as they are cropped from full-body photos.
- **Benign vs Malignant:** Visually, benign moles and malignant melanomas can look quite similar, especially in early stages. For example, some melanomas may only show subtle irregularities in color or border. The dataset includes examples of both, helping the model learn these subtle distinctions.
- The image samples (shown in the placeholder) illustrate the diversity: lesions vary in size, shape, color, and skin tone of the patient. This diversity is important so our model can generalize across different real-world scenarios and patient populations.
  [Insert sample lesion images (benign vs. malignant) here]

Speaker Notes:
This slide would normally show a few example lesion images from the SLICE-3D dataset (we have a placeholder here). These sample images give an idea of what the model sees. Because these are taken from total-body photography, they're not the neat dermoscopic close-ups – they look more like what a person might capture with a phone. We have a mix of **white-light images** (normal illumination) and **cross-polarized images** which reduce glare and can show vascular structures. You can also notice background skin in some crops, since lesions come from larger photos. The key point is how **variable** these images are:

lighting changes, different parts of the body, etc. On the right, for instance, imagine one image of a benign mole and another of a melanoma – to an untrained eye they might both just look like irregular spots. Melanomas might have slightly more uneven coloration or an irregular border, but it's subtle. By including many such examples, the dataset teaches the AI to pick up on those subtle clues. Diversity in skin tones, lesion sizes, and imaging types makes the model more robust for real-world use, where every case looks a bit different.

Slide 5: Challenges in Model Development

- **Extreme Class Imbalance:** Malignant cases are < 0.1% of the dataset (~393 cancer vs ~400k benign) [8]. This imbalance risks the model simply predicting "benign" for everything. Standard training can be biased toward the majority class, making it hard to detect the rare positives (which are clinically most critical).
- **Varied Image Sizes & Quality:** Lesion crops from total-body images came in inconsistent resolutions and aspect ratios [9]. Also, real-world images have diverse lighting, backgrounds, and patient skin tones [10]. Without careful preprocessing, this variability would hinder the model's ability to learn generalizable features.
- **Limited Positive Samples & Overfitting:** With so few melanoma examples, there's a high risk of overfitting – a complex model might just memorize those few malignant lesions instead of learning general patterns [11]. We needed strategies to prevent the model from over-specializing on the training examples and to ensure it performs well on new, unseen data.

Speaker Notes:
Before designing our solution, we faced several key challenges. **First, class imbalance:** Out of ~401k images, only a few hundred are malignant [8]. That's an imbalance of about 1000:1. A naïve model would be overwhelmed by benign examples and could achieve 99.9% "accuracy" by always predicting benign – obviously unacceptable because missing a melanoma is dangerous. We needed to address this imbalance to make the model sensitive to the malignant cases. **Second, variability in the input images:** Because these lesions come from different clinics and cameras, the image sizes weren't uniform. One crop might be 100×100 pixels, another 500×500, etc., plus differences in aspect ratio [9]. We also have variations in lighting (some are brighter, some have cross-polarization), different skin colors, and so on [10]. Neural networks expect consistent input dimensions and can struggle with such heterogeneity, so we had to normalize and augment the data to help the model cope with these differences. **Third, limited positive samples:** With under 400 melanomas to learn from, complex models could easily overfit [11]. For instance, a model might "remember" a particular melanoma image rather than learning the features that make it cancerous. Overfitting would mean great performance on training data but poor detection on new patients – which we must avoid. These challenges guided our approach: we put a lot of thought into data balancing, preprocessing, and choosing model architectures appropriate for the data scale.

Slide 6: Our Approach (Transfer Learning & Metadata Fusion)

- **Transfer Learning on CNNs:** We fine-tuned state-of-the-art convolutional neural network backbones pre-trained on ImageNet – specifically EfficientNetV2-B0, EfficientNetV2-B2, ResNet-50, and MobileNetV3-Large [12]. Using pretrained models gave us a strong starting point for feature extraction, critical given our limited malignant training samples.
- **Multi-Modal Fusion:** Our model ingests both image and metadata. The lesion image passes through a CNN (e.g. EfficientNet) to produce a feature vector, while simultaneously the 44 patient/

lesion metadata features are encoded (via one-hot and normalization) into a 71-dimensional vector [13]. These two representations are then concatenated in the network, allowing the model to learn from visual patterns **and** patient context together for the final malignancy prediction.

- **Imbalance Mitigation:** We implemented class rebalancing and weighting strategies. During training, we under-sampled benign images and over-sampled malignant cases to create a more balanced batch distribution [14]. Additionally, a class-weighted loss function was used to penalize errors on the minority class more, helping the model pay attention to the rare malignancies.

Speaker Notes:
To tackle the challenges, our approach has three main pillars. **First, transfer learning:** Instead of training from scratch, we took advantage of powerful CNN architectures that were pre-trained on large datasets like ImageNet. We used EfficientNetV2 (both B0 and B2 variants), ResNet-50, and MobileNetV3-L [12]. These are well-known models: EfficientNets are optimized for accuracy and efficiency, ResNet-50 is a deep residual network, and MobileNetV3 is designed to be lightweight. By fine-tuning these networks on our skin lesion data, we leveraged their learned visual features (like edge detectors, textures) which is crucial since we have relatively few cancer examples. **Second, metadata fusion:** A key novelty of our approach is incorporating patient metadata into the prediction. We don't rely on images alone – we also input things like the patient's age, sex, lesion location, etc. We preprocess these 44 metadata fields into a fixed 71-dimensional vector (e.g., one-hot encoding categories, normalizing numeric values) [13]. In our model architecture, we have two input branches: one for the image (through a CNN) and one for the metadata (through a series of dense layers). The outputs of these branches are concatenated, so the model's later layers see a combined feature vector. This way, the model can learn correlations like "an irregular dark lesion + patient is older + lesion on back = higher risk." **Third, handling class imbalance:** To prevent the model from ignoring melanomas, we balanced our training process [14]. We downsampled the overwhelming benign class and augmented/duplicated malignant cases during training. We also applied class weighting in the loss function to make each melanoma count more in the training objective. These techniques ensure the model gets sufficient exposure to malignant examples and treats those errors as more costly, which is what we want in a medical setting (missing a cancer should be heavily penalized).

Slide 7: Data Processing Pipeline (Preprocessing & Augmentation)

- **Image Preprocessing:** Lesion crops were extracted and then resized to a uniform resolution (we standardized to 224×224 pixels for most models) [15]. This fixed input size is required for batch training and CNN architectures. We also applied color normalization (adjusting brightness and contrast) so that images have consistent intensity distribution, mitigating differences in lighting across photographs [15].
- **Augmentation:** To boost effective data size and robustness, we performed on-the-fly augmentations on training images. Each image had a 50% chance to undergo random horizontal flip, rotations, scaling (zoom-in/out), or contrast/color jitter [16] [17]. These augmentations mimic real variations (like a lesion photographed from a different angle or under different lighting) and help the model generalize to new images. (Validation and test images were **not** augmented, to evaluate true performance.)
- **Metadata Encoding:** We transformed 44 metadata features into model-ready inputs [18]. Categorical fields (e.g. sex, lesion location) were one-hot encoded, and continuous fields (e.g. lesion diameter) were normalized. After encoding, each lesion's metadata becomes a 71-dimensional feature vector [19]. This pipeline ensures the tabular data is clean and scaled properly to be combined with image features during training.

Speaker Notes:
Here we outline our data pipeline. For **images**, the first step is resizing. We take each lesion crop and scale it to a standard 224×224 pixel size (EfficientNet and ResNet expect ~224 px inputs) [15]. Some experiments also used 128×128 (for faster prototyping), but 224 was used for final models. We maintain aspect ratio by cropping if needed, since slight distortions are okay given augmentation will vary them. We also normalize pixel values, essentially adjusting brightness and color so that one image isn't inadvertently brighter or darker than another – this helps remove lighting bias [15]. Next, **data augmentation**: this is crucial given we have so few melanomas. We programmatically apply random transformations to each training image per epoch [16]. For instance, an image might be randomly flipped horizontally, rotated a few degrees, scaled up or down, or have its contrast changed – each with 50% probability [17]. Over many epochs, the model might see slightly altered versions of the same lesion, which helps it learn the core features rather than memorizing exact pixels. It also simulates the kind of variability we'd see if different people took a photo of the same mole. Importantly, we do **not** augment validation/test images so that our evaluation remains fair. On the **metadata** side, we processed each of the 44 features from the dataset [18]. For example, "anatom_site_general" (body location) which has categories like head/neck, torso, etc., becomes multiple binary columns (one-hot encoding). Numeric features like age or lesion size are scaled to a standard range (like 0 to 1). After doing this for all fields, each lesion's metadata is represented as a vector of length 71 [19]. We integrate this into the data pipeline so that for each image fed to the model, we have a corresponding standardized metadata vector fed alongside it.

Slide 8: Model Architecture Overview [PLACEHOLDER]

- **Dual-Input Network:** Our architecture has two branches: one for the image and one for metadata. The image branch uses a pretrained CNN (EfficientNet, ResNet, etc.) to convert the 224×224 lesion image into a high-level feature map, which is then pooled into a feature vector [20]. The metadata branch is a smaller fully-connected network that takes the 71-dim metadata vector and processes it through dense layers to encode the patient/context information.
- **Feature Fusion:** The outputs of the two branches – the visual feature vector and the metadata feature vector – are concatenated into one combined representation [21]. This fused vector is fed into subsequent dense layers which learn to interpret combined patterns (e.g., certain image features might be more indicative of malignancy in older patients). Finally, a sigmoid activation outputs the probability of the lesion being malignant.
- **Design Rationale:** By fusing image and tabular data, the model can leverage both visual cues and patient context simultaneously. This architecture is aimed at maximizing predictive performance: the CNN handles complex image pattern recognition, while the metadata branch incorporates known risk factors, making the overall prediction more robust and clinically relevant.

Speaker Notes:
This slide (with a diagram placeholder) illustrates our **multi-modal network architecture**. On the left, we have the image input going through a convolutional neural network – for example, EfficientNetV2-B0. This CNN has many layers of convolution and pooling that progressively abstract the image into high-level features. Ultimately, the CNN outputs a vector (after global pooling) that might be, say, 1280 features long for EfficientNetB0. On the right, we have the metadata input. That 71-dimensional vector (with age, sex, etc.) goes through a few dense (fully connected) layers. Think of these as extracting relevant signals from the metadata (for instance, the network can weight features like "male vs female" or certain lesion locations if they matter for melanoma risk). Now, the **fusion**: we take the output vectors from both branches and concatenate (merge) them [21]. The merged vector could be roughly 1400 features (1280 from image + 120

or so from metadata after its dense layers, depending on architecture). This combined representation now contains everything the model has extracted about the lesion's appearance and context. We feed that into a final series of dense layers, which produce a single output neuron that gives a probability of malignancy (using a sigmoid activation for binary classification). We trained this end-to-end, so the CNN and metadata branch both learn jointly to minimize the loss. The rationale here is powerful: Certain visual patterns might only be concerning if, say, the patient is older, or if the lesion is on an atypical body part. By giving the model both kinds of data, we allow it to learn complex rules like "a dark lesion + patient >50 years old + on the scalp = high risk." Pure image models can't do that. So this architecture is both novel and practical, combining deep image analysis with structured clinical data.

Slide 9: EfficientNetV2-B0 Model [PLACEHOLDER]

- **EfficientNetV2-B0:** We utilized EfficientNetV2-B0 as one of our image backbones. B0 is the smallest variant of EfficientNetV2, with ~5.9 million parameters and a 1280-dimensional feature output [22]. It's known for its optimized architecture that achieves strong accuracy with relatively low computational cost [23] – an advantage for deployment on limited hardware.
- **Performance:** EfficientNetV2-B0 emerged as our top-performing model. It rapidly learned the lesion classification task, reaching a validation ROC AUC of ~0.93 (92.9%) at peak [24]. Notably, B0's training curve showed a steady, stable improvement to this high AUC within ~7 epochs [25]. This suggests that despite being a lightweight network, B0 captured the critical features of melanoma effectively. Its combination of high accuracy and efficiency makes it well-suited for real-world screening applications.

Speaker Notes:
On this slide, we highlight **EfficientNetV2-B0**, which was one of the four CNN architectures we tried – and it turned out to be the best. EfficientNetV2-B0 is relatively small: about 5.9 million parameters [22] (compare that to ResNet-50's 25 million). EfficientNets are designed through a compound scaling strategy – basically, they scale depth, width, and image size in a balanced way which gives them an edge in both accuracy and efficiency. We leveraged a pretrained B0 and fine-tuned it on our data. The diagram (placeholder) would show B0 feeding into our fusion architecture. In terms of results, B0 was remarkable. It **achieved the highest AUC of all models, roughly 0.929** or ~92.9% [24] on the validation set. This means it was very accurate at ranking lesions by risk. The learning curve for B0 was also very smooth – by about epoch 6–7 it already plateaued at that high AUC [25]. This stability indicates it didn't overfit badly; it generalized well to validation data. Given it's the smallest model we tried, this outcome is interesting – it suggests that the EfficientNet architecture is very well-suited to this task, extracting just the right features without excessive complexity. For a deployment perspective, B0's efficiency (fewer parameters, faster inference) plus its top accuracy is a big win. It could feasibly run on a smartphone or a small clinic device to screen patients in real time.

Slide 10: EfficientNetV2-B2 Model [PLACEHOLDER]

- **EfficientNetV2-B2:** A larger EfficientNetV2 variant we tested, with ~8.8 million parameters and a 1408-dimensional feature output [26]. B2 has more capacity (more layers/width) than B0, potentially capturing more complex patterns. It retains the EfficientNet family's compound-scaled design for balanced accuracy vs. efficiency.
- **Performance:** EfficientNetV2-B2 achieved the second-best results in our experiments. It reached a validation AUC of ~0.92 (92.0%) [24], very close to B0. B2 learned quickly as well – hitting its peak

performance around epoch 5. However, it plateaued slightly earlier than B0. This could be due to needing more training or fine-tuning to fully utilize its extra capacity. Regardless, B2's high AUC confirms the strength of EfficientNetV2 architecture, with B2 providing nearly as good accuracy as B0 while being a bit larger in size.

Speaker Notes:
Now, EfficientNetV2-B2 was the bigger sibling of B0 that we tried. With around 8.8 million parameters [26], it has more layers and filters, so in theory it can capture more detailed features. It also outputs a larger feature vector (1408 dims from the CNN). We expected B2 might outperform B0 due to this greater capacity. In practice, **B2 came in a very close second place**. It achieved about **0.920 AUC** on validation [24], basically 92.0%, just a hair under B0's 92.9%. The training curve for B2 showed that it improved really fast and actually peaked by around the 5th epoch at that ~0.92 AUC. After that, it didn't gain much more – possibly with more training epochs or some hyperparameter tweaks, B2 could match or exceed B0. One hypothesis is that because B2 has more parameters, it might require a bit more regularization or a longer training schedule to fully shine, otherwise it could plateau early. Still, B2's performance is excellent and it validates that EfficientNetV2 models (both B0 and B2) are exceptionally effective for this problem. The difference between them was small; B0 slightly edged out B2 in our timeframe. From a deployment standpoint, B2 is slightly heavier computationally, so one might choose B0 for speed or B2 if a bit more accuracy can be squeezed out with further tuning.

Slide 11: ResNet-50 Model [PLACEHOLDER]

- **ResNet-50:** A classic deep CNN with 50 layers and ~25 million parameters. It employs *residual learning* (skip connections) to ease training of such a deep network. ResNet-50 is a proven architecture for image feature extraction, capturing hierarchical features through its stacked convolutional blocks [27].
- **Performance:** In our study, ResNet-50 underperformed compared to EfficientNets. Its best validation AUC was only around ~0.69 (≈69%) [28] – the lowest of the four models. We observed signs of overfitting: ResNet initially learned (training accuracy improved) but validation AUC stagnated or declined. The model's large capacity did not translate well given the limited melanoma data [29], indicating it was too complex and prone to memorizing noise or benign-class bias.

Speaker Notes:
ResNet-50 has been a workhorse in computer vision for years – it introduced the idea of skip connections which allow training very deep networks. With 25 million parameters, it's much larger than EfficientNetB0 or B2. We brought ResNet-50 into our experiment expecting it to learn very rich features (it's good at capturing fine details and complex textures). However, **ResNet-50's performance was disappointing in this project**. It achieved only about **0.69 AUC on validation** at best [28]. This is significantly lower than the ~0.92 we saw with EfficientNets. What happened? Essentially, ResNet-50 struggled with our data. We suspect overfitting: it likely memorized aspects of the training lesions that didn't generalize. Remember we have just 393 malignant examples – a network this large can effectively memorize many of them. We saw the training metrics for ResNet going up, but validation not really following – a classic sign that it wasn't generalizing. The model's "larger capacity did not translate to better performance" in our setting [29]. Possibly, without more data or stronger regularization, ResNet-50 was just too complex. Another factor: ResNet was designed in 2015; architectures like EfficientNet are newer and more efficient. So, in summary, ResNet-50, while powerful in theory, ended up being the weakest model for our task, likely because it overfit and had difficulty handling the severe class imbalance and variability without additional tuning.

Slide 12: MobileNetV3-Large Model [PLACEHOLDER]

- **MobileNetV3-Large:** A modern mobile-optimized CNN with ~5 million parameters. It uses depthwise separable convolutions and network architecture search optimizations to achieve high efficiency. MobileNetV3 is lightweight and fast, intended for deployment on devices with limited computing (e.g. smartphones) [30] .
- **Performance:** MobileNetV3 had the fastest inference but achieved only moderate accuracy on our task. Its best validation AUC was around ~0.78 (78%) [28] . Notably, it reached that peak early in training but then failed to improve further – and even slightly declined, indicating possible underfitting or instability. The "ultra-small" model capacity [31] likely missed some complex features, resulting in lower discriminative power compared to EfficientNet. In practice, MobileNetV3's low compute needs are great for speed, but the trade-off was a significant drop in accuracy in detecting melanoma.

Speaker Notes:
MobileNetV3-Large is at the opposite end of the spectrum from ResNet in terms of size – about 5 million parameters, even a bit fewer than EfficientNet-B0. It's designed with mobile devices in mind, meaning it's very computationally efficient (using tricks like separable convolutions and optimized block structures). We included MobileNetV3 to see how a very lightweight model performs, thinking it could be useful for a smartphone app. The result was **somewhat expected: MobileNetV3 had the lowest accuracy of the four models aside from ResNet**. It peaked at roughly **0.78 AUC** [28] . Interestingly, it hit the high 70s AUC within the first couple of epochs and then plateaued or even got a bit worse. This pattern suggests it might have underfit – basically, it learned the easy distinctions quickly but its limited capacity prevented it from learning the harder, subtle patterns. The training was a bit unstable too; small models can sometimes oscillate or struggle with noisy gradients (especially given class imbalance). In plain terms, MobileNetV3 just wasn't complex enough to capture all the features needed to separate melanomas from benign lesions reliably. It gave okay performance (around 78% AUC means it's better than random, but far from excellent). The **advantage of MobileNet** is its speed and low resource usage [30] – it could run on a phone very smoothly. But our findings highlight the trade-off: that efficiency came at the cost of significantly lower accuracy. For a critical application like cancer screening, that drop in accuracy might not be acceptable unless we improve it or use it in an ensemble with other models.

Slide 13: Results – AUC Curves & Model Performance [PLACEHOLDERS]

- **Training AUC Curves:** The learning curves (see graph) highlight that EfficientNetV2-B0 (blue) and B2 (orange) not only started strong (AUC ~0.6–0.7 on first epoch) but quickly surpassed 0.90 AUC within a few epochs [25] . Their curves climbed steadily with minimal overfitting. In contrast, ResNet-50 and MobileNetV3 (dashed lines) plateaued much lower; MobileNet even peaked early and then dipped, showing unstable training.
- **Best Validation AUCs:** EfficientNetV2-B0 achieved the highest AUC ≈ **0.929** (92.9%), with B2 slightly behind at **0.920** [32] . MobileNetV3-Large reached ~**0.78** AUC and ResNet-50 only ~**0.69** [32] . *(Bar chart on the right visualizes these peak AUC values for easy comparison.)* The EfficientNet models clearly outperformed the others by a wide margin.
- **Model Stability:** EfficientNet V2 models showed more stable and consistent training, maintaining improvement each epoch [33] . ResNet-50 and especially MobileNetV3 struggled – their validation performance stalled or degraded, reflecting difficulties in learning or generalization after initial

epochs. This reinforces that architecture choice (and handling of data imbalance) strongly affected outcomes.
[Insert training & validation AUC curves plot here]
[Insert bar chart of best AUC for each model here]

Speaker Notes:
This slide summarizes our results visually. On the left, imagine a plot of **validation AUC over training epochs for each model**. You would see EfficientNetV2-B0 and B2 rising quickly into the ~0.9 range [25], whereas ResNet-50 and MobileNetV3 languish much lower. B0's curve in particular goes up smoothly and plateaus around 0.93 AUC, and B2 around 0.92, by about epoch 5–7. ResNet-50 maybe plateaus around 0.65–0.69 and doesn't improve much after that – indicating it's not learning further or possibly overfitting early. MobileNet might actually go up to ~0.78 in epoch 2 or 3, then slightly go down or flatline, showing it hit capacity quickly and even got worse as training continued (perhaps due to overfitting to majority class, etc.). On the right side, we have a bar chart with the **final/peak AUCs**: you can clearly see B0 ~92.9%, B2 ~92.0%, MobileNet ~78%, ResNet ~69% [32]. The gap is striking – roughly a 15–25 percentage point difference in AUC between EfficientNets and the older/smaller models. In terms of stability, our observations were that **EfficientNet models trained very consistently** [33] – they didn't show signs of overfitting or performance collapse, likely due to their better regularization and the fact we stopped at optimal epochs. ResNet and MobileNet were trickier: ResNet might have needed more regularization or differently configured training, and MobileNet probably needed more capacity or an ensemble to reach higher accuracy. The takeaway is that choosing a modern, high-performing architecture (EfficientNetV2) was key to our success, and the data augmentations + transfer learning likely helped those models shine whereas the others still fell short.

Slide 14: Model Comparison & Discussion [PLACEHOLDER]

- **Accuracy vs. Complexity:** EfficientNetV2-B0 and B2 achieved the highest AUCs (~93% and 92%) while using relatively modest model sizes (6–9 million params) [24]. This indicates an excellent trade-off: these models are both accurate and efficient. In contrast, ResNet-50 had ~25M params but yielded only ~69% AUC – complexity alone did not help [34].
- **ResNet-50:** Large capacity, deep architecture, yet underperformed due to overfitting and the data constraints. Its poor showing (AUC < 0.70) underscores that more parameters can be a liability without sufficient data or proper regularization [29].
- **MobileNetV3:** Smallest model (~5M params) and fastest, but it underfit the problem (AUC ~0.78) [35]. It likely missed finer features needed for distinguishing melanomas. MobileNet's efficiency is appealing, but a ~15% absolute AUC drop relative to EfficientNet is a significant performance trade-off.
[Insert table comparing models – Params, AUC, notes, etc. here]

Speaker Notes:
This table (placeholder) distills the comparison of the four models. We list each model, number of parameters, and their best validation AUC, along with a note or two on their performance characteristics. The key pattern is **clear**: the EfficientNetV2 models (B0 and B2) are both **highly accurate and reasonably compact** [24]. B0 (5.9M params) and B2 (8.8M params) achieved ~93% and ~92% AUC respectively, far outperforming the others. ResNet-50 is about 3× the size of B2 (25M params) but only got to ~69% AUC [34] – a classic case of diminishing returns and even negative returns on using a bigger model without enough data. It overfit and didn't generalize. MobileNetV3, on the other hand, is slightly smaller than B0 (around 5M params) and indeed was computationally efficient, but it only reached ~78% AUC [35]. The ~15 point drop

from EfficientNet suggests that MobileNetV3's lightweight design sacrifices some representational power, which hurt in a task as nuanced as melanoma detection. In practice, if we were choosing a model to deploy, **EfficientNetV2-B0 stands out as the best choice**: it has the top accuracy and is still lightweight enough for practical use. B2 is close behind if we can afford a bit more compute. ResNet-50 would be ruled out due to low accuracy, and MobileNetV3 might be too unreliable despite its speed. This comparison highlights how newer architecture (EfficientNetV2) gave us a big leap in performance, and also that balancing model complexity to dataset size is crucial – too big (ResNet) or too small (MobileNet) both performed worse than the "right-sized" EfficientNets.

Slide 15: Conclusion & Future Work

- **Conclusion:** We demonstrated that a deep learning model can accurately detect skin cancer (melanoma) from smartphone-quality images when trained on the SLICE-3D dataset (combining lesion photos + metadata). Our best model (EfficientNetV2-B0 + metadata) achieved ~**93% AUC**, approaching dermatologist-level discrimination [36]. Incorporating patient metadata alongside images improved the model's context-awareness, boosting its ability to distinguish benign vs malignant lesions in a realistic setting.
- **Impact:** This work suggests AI-driven melanoma screening is feasible for broad use. A model like ours could serve as a pre-screening tool to flag suspicious lesions for further examination, potentially extending early detection to primary care or remote regions. By leveraging ordinary total-body photos, our approach can make screening more accessible, addressing gaps in dermatology access [37] and aiding clinicians with a reliable second pair of eyes.
- **Future – Improving the Model:** We plan to expand the training data, especially adding more confirmed melanoma cases (or augmenting them) to further improve generalization [38]. Exploring **model ensembling** (e.g., combining EfficientNet-B0 and B2) and performing more extensive hyperparameter tuning could yield additional performance gains [39]. We also consider training specialized sub-models (image-only or metadata-only) to understand each modality's contribution and to serve as backups if one data type is missing.
- **Future – Deployment:** Before real-world deployment, the model needs validation on fresh, real-world user data (e.g., photos taken by patients) [40]. We aim to integrate the system into a user-friendly mobile or web application for clinicians and patients. This involves adding features like probability calibration and explainability (e.g., Grad-CAM visualizations) to increase trust. Ultimately, a tool arising from this work could be tested in prospective clinical trials to evaluate its impact on early melanoma detection and patient outcomes.

Speaker Notes:
To conclude, our project showed that **deep learning can be effectively applied to skin cancer detection using real-world images**. We reached a high accuracy – our best model got about 93% AUC on validation, which is quite impressive and in the ballpark of expert performance for melanoma screening [36]. One reason for this success was that we didn't treat it as just an image problem; we fed the model metadata like patient age and lesion location, which made it smarter and more context-aware. This resulted in a system that can look at a regular photo of a skin spot and give a pretty reliable risk assessment. The implications are significant: such an AI tool could be used by general practitioners or even patients themselves as an early warning system. Imagine a smartphone app that you can use to take a photo of a mole and get a risk score – it wouldn't replace a dermatologist, but it could tell you "hey, this looks suspicious, get it checked." This could **extend early detection** to people who live far from dermatology clinics or reduce the load on specialists by filtering out clearly benign cases [37].

Looking ahead, there are several ways to **enhance and deploy** our work. First, we want to **expand the dataset** – more images, especially more melanoma examples, possibly by incorporating other public datasets or collecting new images [38]. More data can help the models generalize better and not miss edge cases. We also consider **model ensembling and fine-tuning**: for instance, combining the strengths of B0 and B2 might give a small boost, and doing a thorough sweep of hyperparameters (learning rates, regularization techniques) could help, especially to get ResNet or others to perform better [39]. Another idea is training image-only vs metadata-only models to quantify how much each data source contributes and to have fallback models if, say, metadata isn't available in some use scenario.

On the **deployment side**, we need to test this model on completely independent, real-world data – perhaps a pilot study where patients use their phone to take lesion images and we see how well the model predicts [40]. We'd likely build a prototype application, which involves not just the model but also a user interface, instructions to take good photos, and importantly, adding an explanation system (like highlighting areas of the image the model finds concerning, via Grad-CAM). That can build trust with doctors and patients, who will want to know *why* the AI is saying "this mole looks risky." We'd also calibrate the probability outputs to make them clinically meaningful – maybe we choose a threshold that yields high sensitivity (catch almost all melanomas) while accepting some false alarms, which is reasonable in a screening context. Ultimately, the goal is to integrate this into clinical workflow and validate that using such a tool actually leads to earlier melanoma detection and better patient outcomes. That's the future direction – taking this promising model from the lab to the clinic. Thank you for your attention.

---

[1] Melanoma: Symptoms, Staging & Treatment - Cleveland Clinic
https://my.clevelandclinic.org/health/diseases/14391-melanoma

[2] Even with more U.S. dermatologists, rural patients may lack access | Reuters
https://www.reuters.com/article/business/healthcare-pharmaceuticals/even-with-more-us-dermatologists-rural-patients-may-lack-access-idUSKCN1LT380/

[3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] Datasciencereport(1).docx
file://file-HEAdoBYgWgHB4LnhhPa2K5