A PROTOTYPE MOBILE APP OF A SKIN LESION CLASSIFIER

BY: ANIKA AREVALO⚛️
JR AI DATA OPS
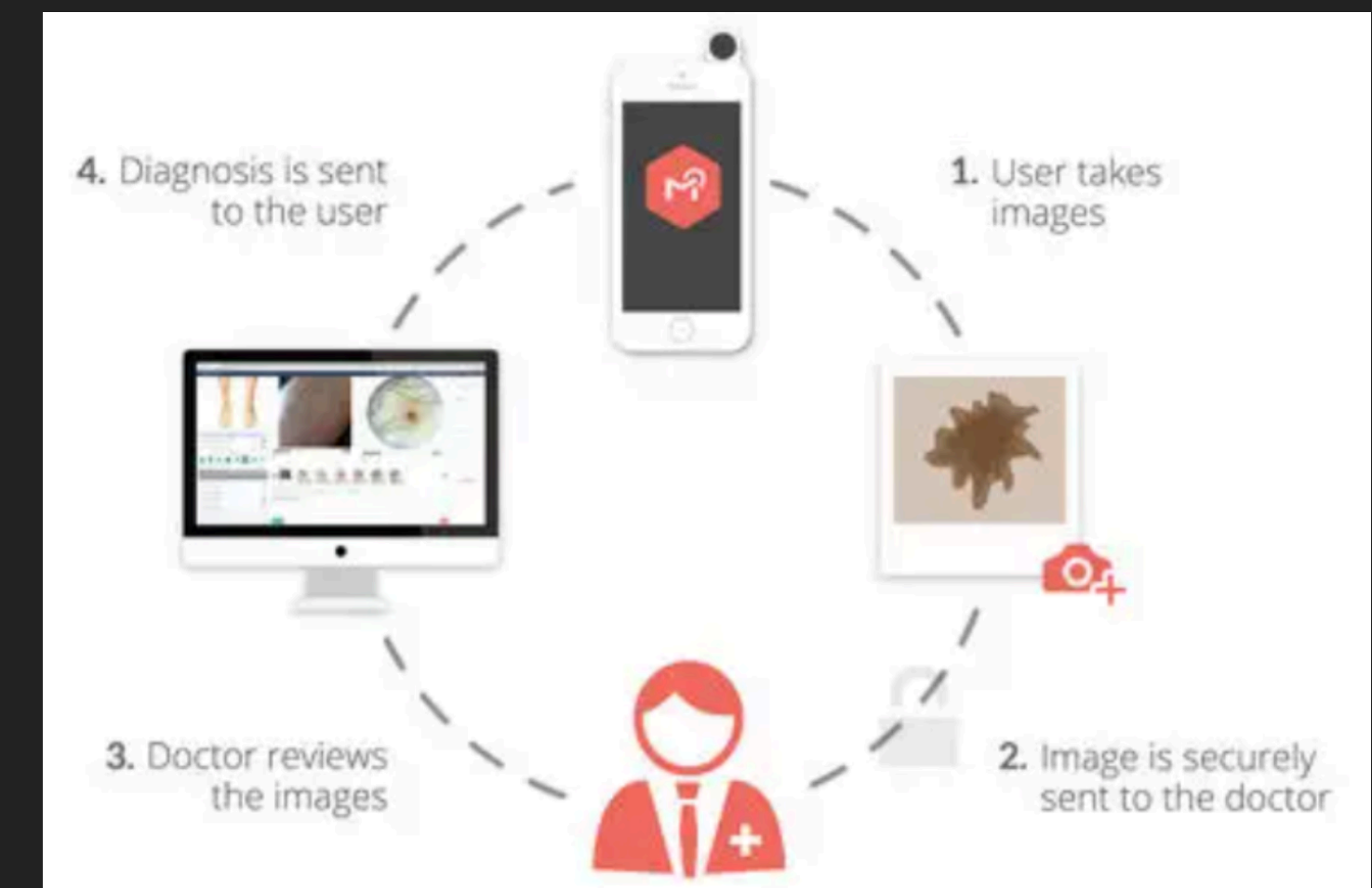### (BECODE, GHENT)

WHY BUILD AN APP TO DETECT SKIN LESIONS?

Skin lesions, which are irregular skin changes in contrast to neighbouring skin surface, can evolve into *skin cancer*, one of the most dangerous cancers.

Melanoma lesions, one of two main skin cancer types, are responsible for the significant increase in mortality and morbidity in recent years; they are the most dangerous among various lesion types.

If the physicians detect skin lesions earlier, they can increase **recovery rate of 90%***

* Hosny, K.M.; Kassem, M.A.; Foaud, M.M. Classification of skin lesions using transfer learning and augmentation with Alex-net. *PLoS ONE* **2019**, *14*, e0217293. [**Google Scholar**] [**CrossRef**] [**PubMed**]

There is a market for **a solution that leverages a Convolutional Neural Network to help people classify different types of skin cancers** *quickly and accurately*.

4. Diagnosis is sent to the user
1. User takes images
2. Image is securely sent to the doctor
3. Doctor reviews the images

THE CATEGORIES OF SKIN LESIONS:

▸ Actinic keratoses and intraepithelial carcinoma **(akiec)**: common non-invasive variants of squamous cell carcinomas. They are sometimes seen as precursors that may progress to invasive squamous cell carcinoma.

▸ Basal cell carcinoma **(bcc)**: a common version of epithelial skin cancer that rarely metastasizes but grows if it isn't treated.

▸ Benign keratosis **(bkl)**: contains three subgroups (seborrheic keratoses, solar lentigo, and lichen-planus like keratoses (LPLK)). These groups may look different but are biologically similar.

▸ Dermatofibroma **(df)**: a benign skin lesion that is regarded as a benign proliferation or an inflammatory reaction to minimal trauma.

THE CATEGORIES OF SKIN LESIONS:

▸ Melanoma **(mel)**: a malignant neoplasm that can appear in different variants. Melanomas are usually, but not always, chaotic, and some criteria depend on the site location.

▸ Melanocytic Nevi **(nv)**: these variants can differ significantly from a dermatoscopic point of view but are usually symmetric in terms of distribution of color and structure.

▸ Vascular Lesions **(vasc)**: generally categorized by a red or purple color and solid, well-circumscribed structures known as red clods or lacunes.

TRANSFER LEARNING FOR IMAGE CLASSIFICATION VIA CONVOLUTIONAL NEURAL NETWORK (CNN)

The entire idea behind Transfer Learning is that you can **take a model that has already been pre-trained on a large dataset, modify it and retrain it on the dataset you're currently working with**.

After doing research on **Convolutional Neural Networks (CNN)**, I became interested in developing an **end-to-end machine learning solution**. I decided to use the **HAM10000 dataset** (which contains approximately 10,000 different images of skin lesions) to build a mobile app to classify skin lesions.
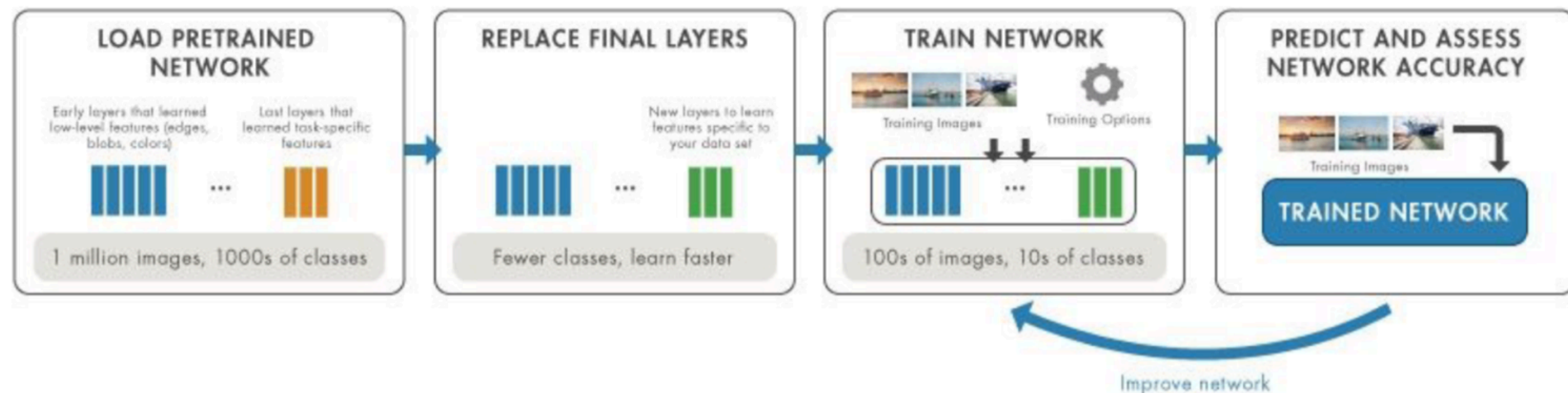
# PRE-TRAINED DEEP LEARNING MODEL FOR MOBILE DEPLOYMENT

For this project, I chose to use the **MobileNet version 2 architecture**, which is optimised for mobile applications with less computing power.
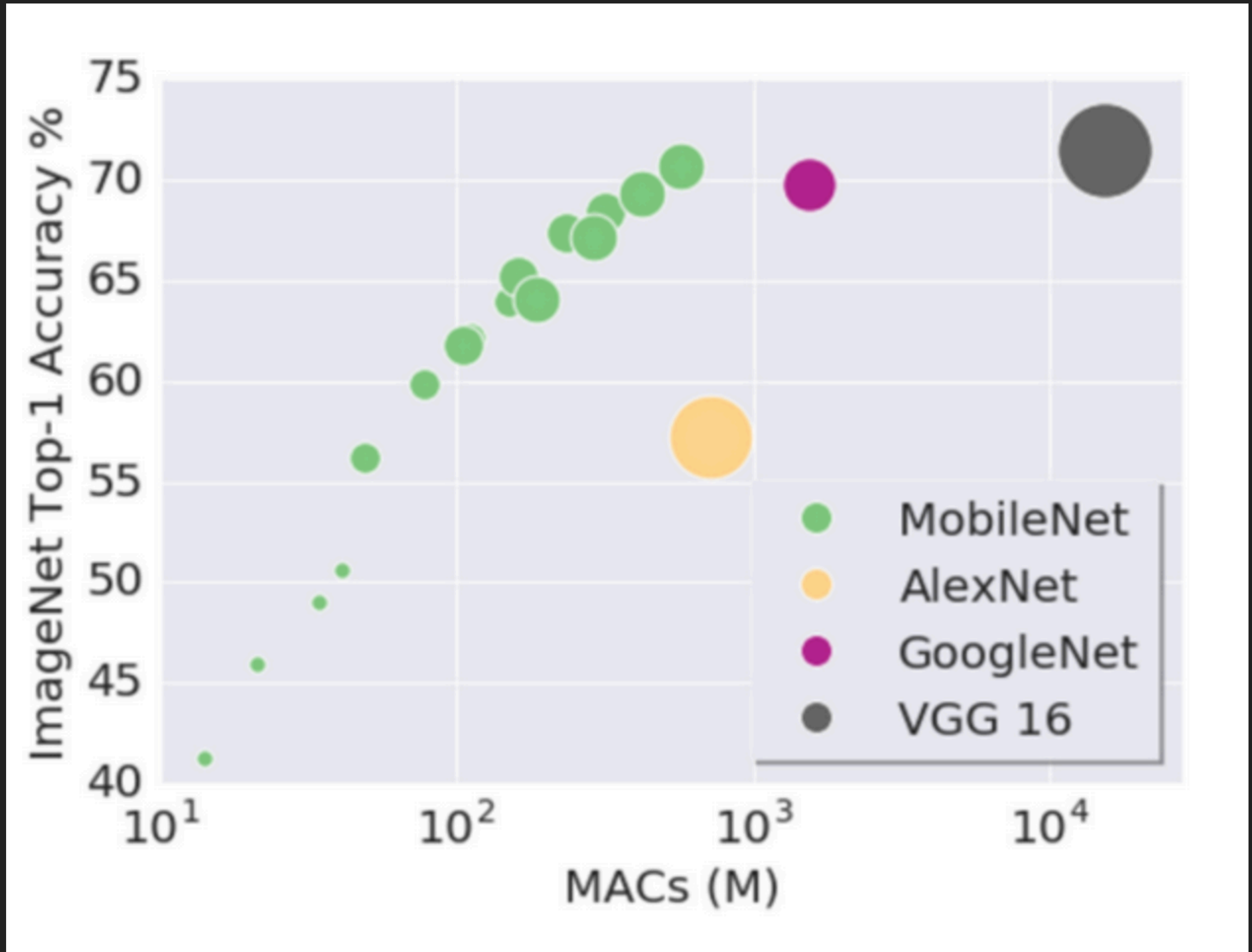
Compared to other popular open-source pre-trained deep learning models

THE CATCH

This bar plot shows that the **distribution of samples is uneven**, and this knowledge is taken into account in the pre-processing phase of the machine learning pipeline.
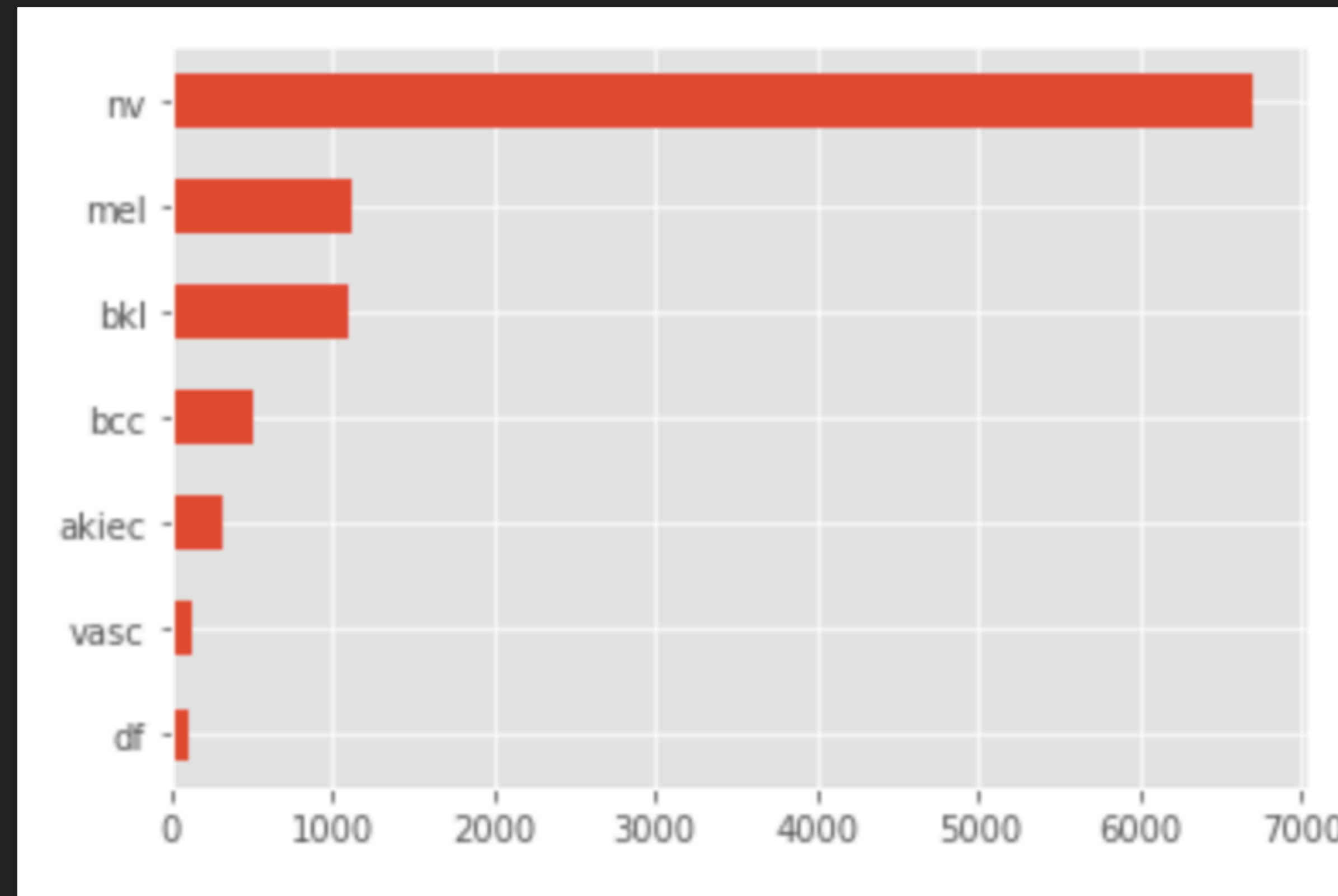
Melanocytic nevi (NV) is around 67% of the entire data set

KEY COMPONENTS OF THIS DEEP LEARNING API PIPELINE

I. **Data Augmentation using Keras** to augment training data during pre-processing;

II. **Keras Callbacks** while compiling the model during training phase in order to capture the 'best' version of the model in training ; and

IIII. The predictive deep learning model **runs locally in the browser or deployed as a mobile app.**

AUGMENTING THE TRAINING DATA

To increase the number of training examples I had to work with, I leveraged data augmentation capabilities of Keras. For this, I used the **Keras ImageDataGenerator class** from the **Keras Preprocessing library**, which generates batches of tensor image data with real-time augmentation by looping through the data in batches.

```python
1  datagen = ImageDataGenerator()
2  aug_data = datagen.flow_from_directory(path,
3                                         target_size=(224, 224),
4                                         batch_size=batch_size)
```

Declaring an augmented data generator by running the following code. The target size is 224x224 because those are the dimensions that are needed for the MobileNetv2 input layer

COMPILING THE MODEL

The **Keras Callbacks library** provides an array of useful functions that can be applied at several stages during the training process of the model. These functions can be used to learn more about the internal states of the model.

Two of the callbacks that are used in this program are **ReduceLROnPlateau** and **ModelCheckpoint**.

REDUCELRONPLATEAU

‣ This is a nifty command to reduce the learning rate when one of the model metrics has stopped improving.

‣ It's been shown that models often benefit when the learning rate is reduced by a factor of 2–10 once the model stops improving after several iterations.

```
1  reduce_lr = ReduceLROnPlateau(monitor='accuracy',
2                                 factor=0.5,
3                                 patience=2,
4                                 verbose=1,
5                                 mode='max',
6                                 min_lr=0.00001)
```

Some parameters that were passed

MODELCHECKPOINT

▸ This is another nifty command used to save the model after every epoch.

▸ **Save_best_only=True** makes sure that the best model isn't overwritten.

```
1   checkpoint = ModelCheckpoint(filepath,
2                                monitor='val_top_3_accuracy',
3                                verbose=1,
4                                save_best_only=True,
5                                mode='max')
```

Saving the best version of the trained model with relative ease

A CONFUSION MATRIX OF THE PREDICTIONS

‣ The **re-trained MobileNetv2 model** has modest performance and that **a significant number of test examples for label nv were classified correctly**.

‣ The model mostly predicts nv (324) and mel (105) correctly, but struggles with akiec and df.

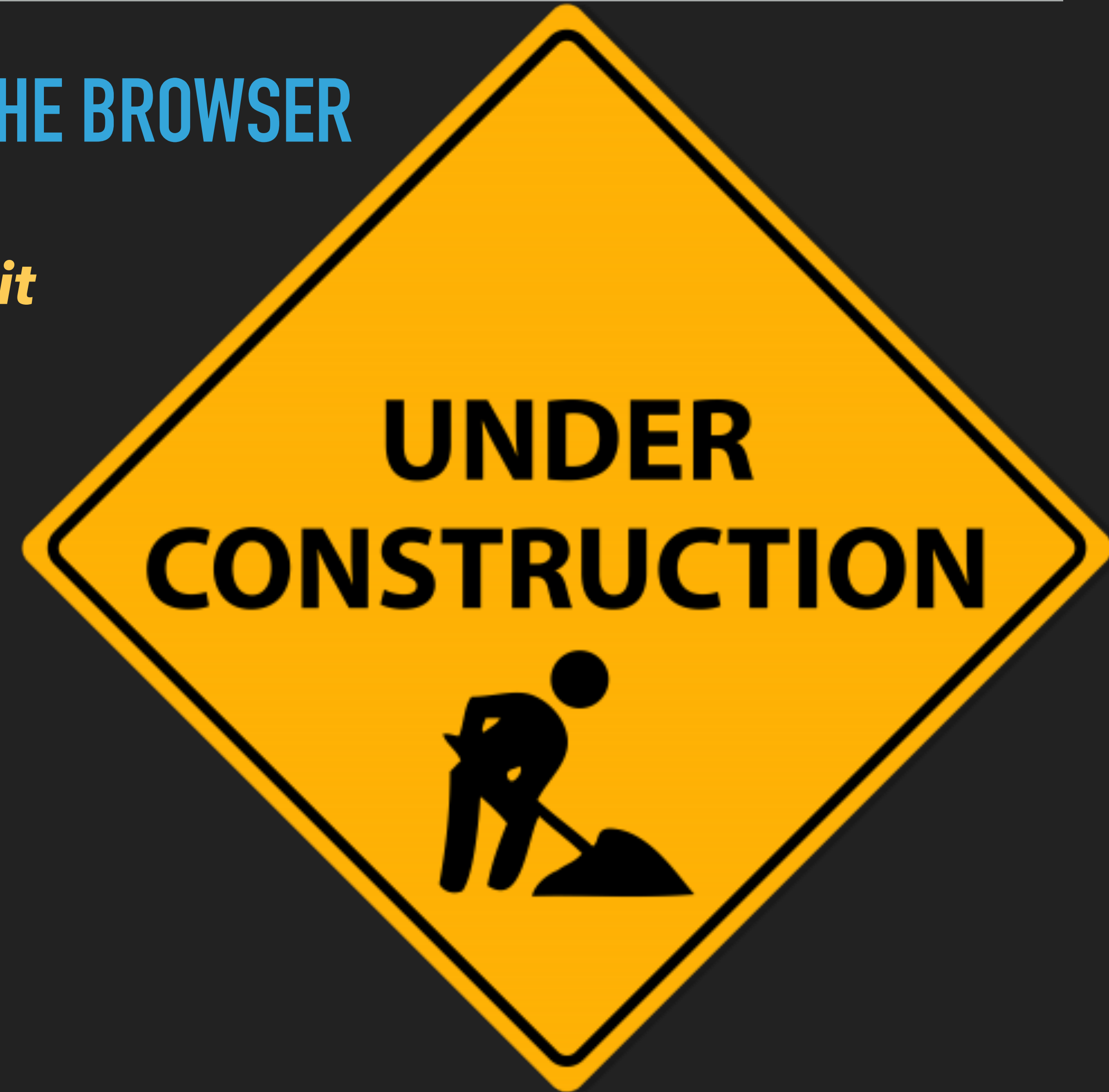‣ The model confuses mel with nv (328), bkl (102), bcc (44) and akiec (32).
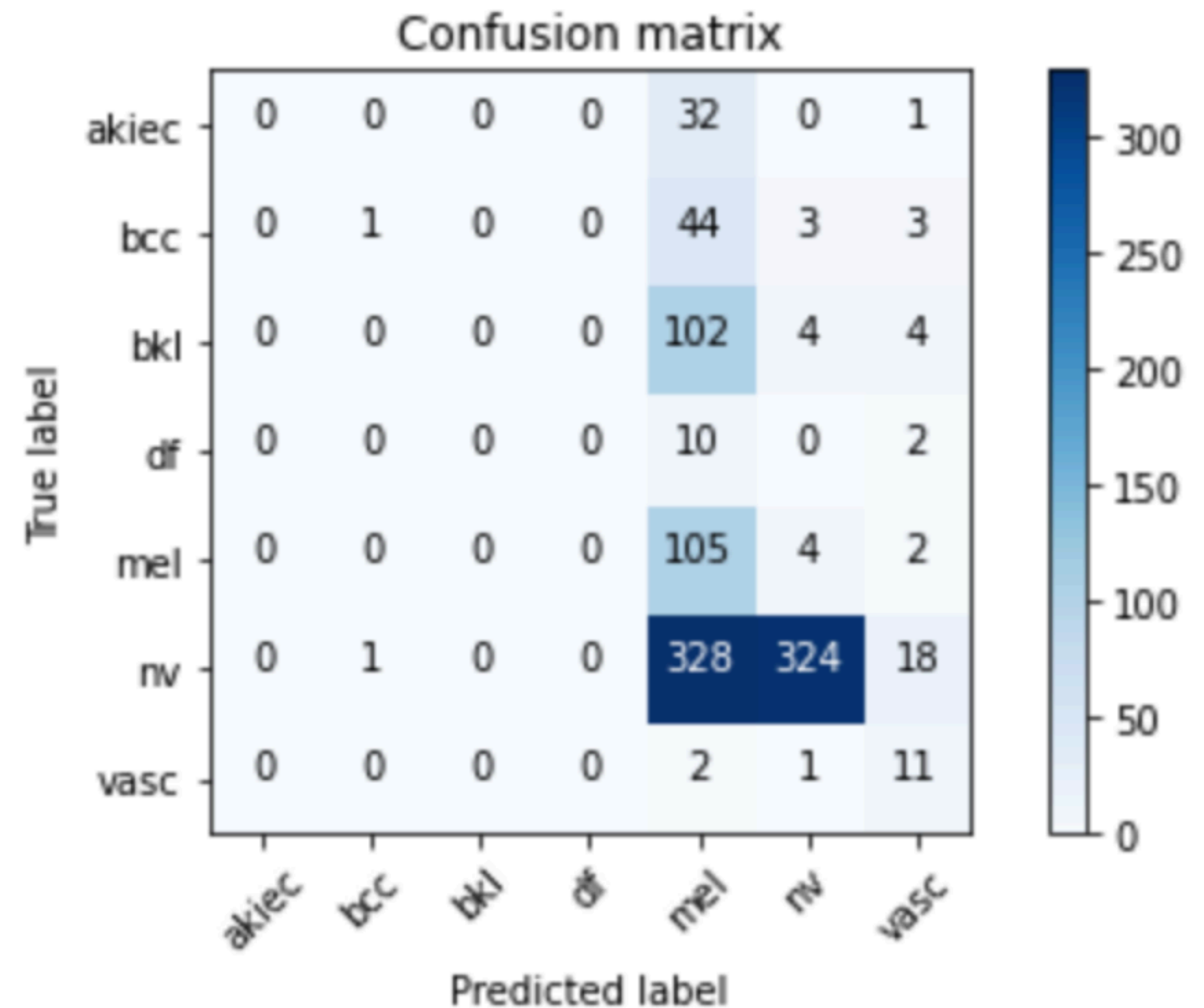
The model still has much more room for improvement

RUNNING THE DEEP LEARNING MODEL IN THE BROWSER

*Plans to deploy the app locally via **Streamlit***

are pending.

'Steal' learning ;)

NEXT STEPS

▸ Availing of a **cloud service** provider for massive computing power and storage capacity

▸ Discovering different techniques to **address class imbalance**

▸ Creating a **bonafide mobile app**

Please feel free to contact me. Let's brainstorm and collaborate!

Anika Arevalo ⚛️

Email: anika.arevalo@gmail.com

LinkedIN: /anika-arevalo

Github: /anikaarevalo

Cat or dog? Persian cat according to ResNet50