Classifying Cat Breeds Using Convolutional Neural Networks:

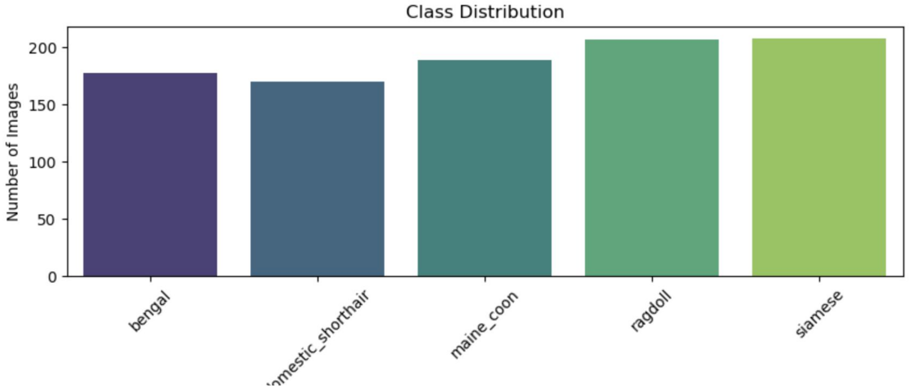
Github Link
May 2024
Linda Ji

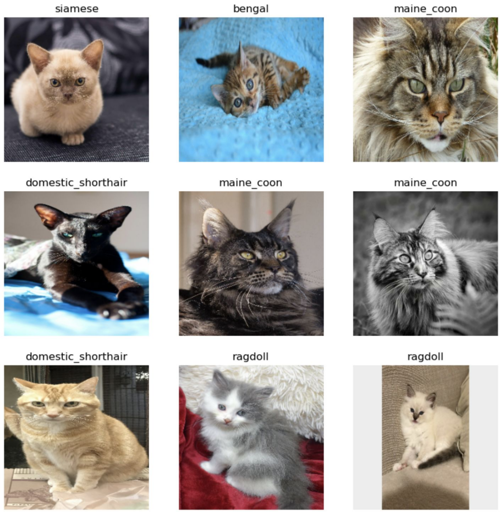
Problem Statement & Data Assumptions

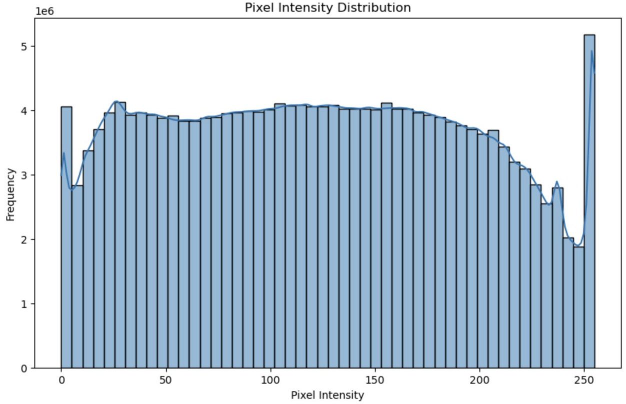
To classify cat images between 5 breeds with a CNN Model

1. Diverse Dataset:
    - Assumption: Dataset is diverse and representative for each breed.
    - Hypothesis: Ensures effective learning of distinguishing features.
2. High-Quality Images:
    - Assumption: Images are high quality with minimal noise.
    - Hypothesis: Improves feature learning and model performance.
3. Consistent Preprocessing:
    - Assumption: Images are consistently resized, normalized, and augmented.
    - Hypothesis: Reduces variability, enhancing model generalization.
4. Sufficient Training Data:
    - Assumption: Adequate training samples for each breed.
    - Hypothesis: Prevents overfitting, ensuring better accuracy on new images

Exploratory Data Analysis







- Total of 951 files, with 5 classes
- Class distribution is relatively uniform, with slight differences
- Pixel distribution is uniform except for extreme bright and dark pixels, indicating normalization of image data would be helpful

Feature Engineering & Transformations

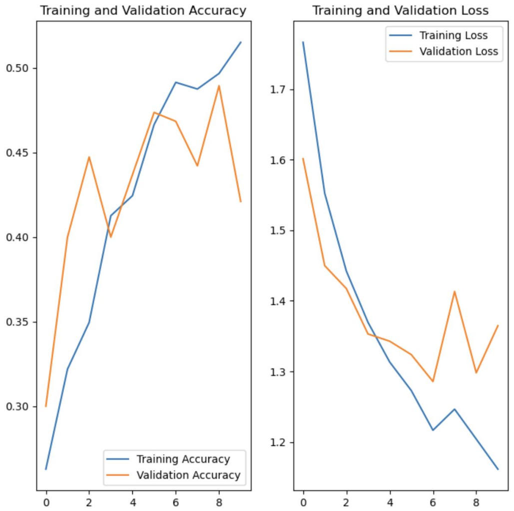
- Resize images
- Split test and train set with seed for reproducibility
- Standardize data
- Data Augmentation

Proposed Approaches (Model) with checks for overfitting/underfitting

Model: "sequential_1"

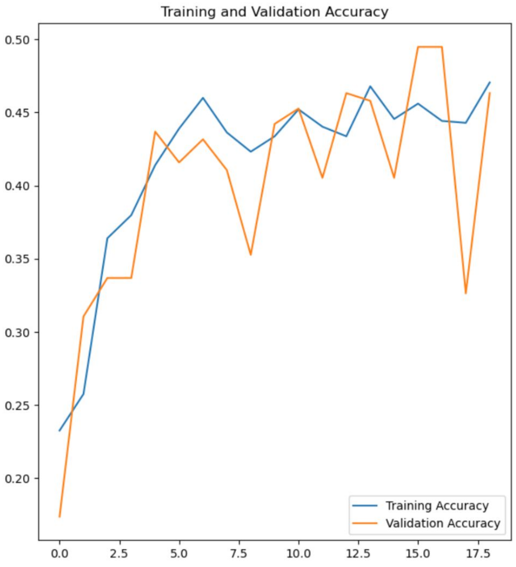
Layer (type)	Output Shape	Param #
sequential (Sequential)	(None, 180, 180, 3)	0
rescaling (Rescaling)	(None, 180, 180, 3)	0
conv2d (Conv2D)	(None, 180, 180, 16)	448
max_pooling2d (MaxPooling2D)	(None, 90, 90, 16)	0
conv2d_1 (Conv2D)	(None, 90, 90, 32)	4,640
max_pooling2d_1 (MaxPooling2D)	(None, 45, 45, 32)	0
conv2d_2 (Conv2D)	(None, 45, 45, 64)	18,496
max_pooling2d_2 (MaxPooling2D)	(None, 22, 22, 64)	0
dropout (Dropout)	(None, 22, 22, 64)	0
flatten (Flatten)	(None, 30976)	0
dense (Dense)	(None, 128)	3,965,056
dense_1 (Dense)	(None, 5)	645

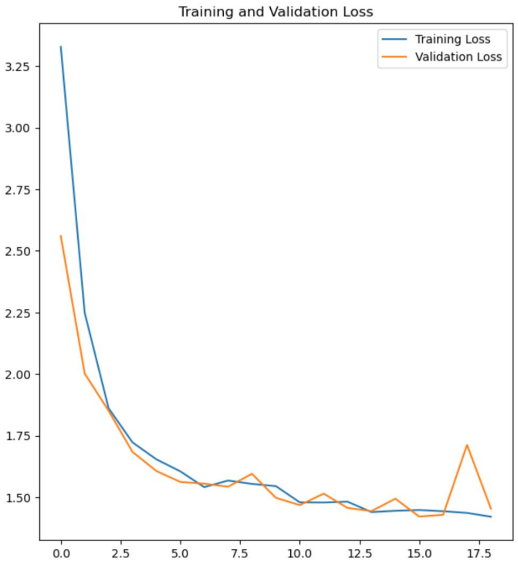
Total params: 11,967,857 (45.65 MB)
Trainable params: 3,989,285 (15.22 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 7,978,572 (30.44 MB)



- For 1st model with data augmentation & drop
- validation loss starts increasing around epoch 6
- Validation accuracy: 0.39 Train accuracy: 0.49, showing overfitting
- This indicates more regularization and early stopping is required to not overfit the model

Proposed Solution (Model Selection) with regularization, if needed





- 3 other models were tried to test the impact of:
- Model 2: regularization + early stopping
- Model 3: Hyperparameter Tuning
- Model 4: Adding more layers to Model 2
- Model 2 showed the highest accuracy, with compared results in next slide

Results (Accuracy) and Learnings from the methodology

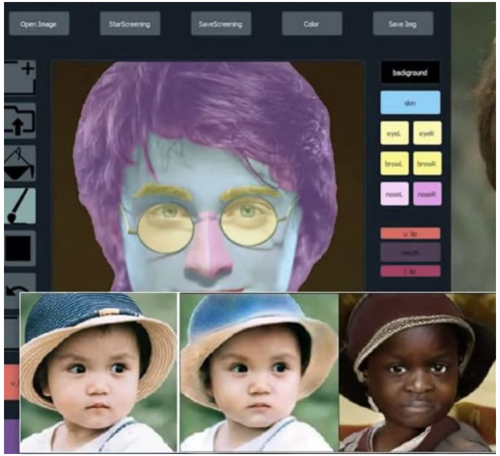
#	Training Accuracy	Test Accuracy	Overfitting	Data Augmentation	Drop out	Regularization	Early Stopping	Hyper parameter Tuning
Model 1	0.4968	0.3949	Yes	Yes	Yes	No	No	No
Model 2	0.433	**0.5261**	**No**	Yes	Yes	Yes	Yes	No
Model 3	0.6761	0.5107	Yes	No	Yes	No	No	Yes
Model 4	0.4794	0.5226	**No**	Yes	Yes	Yes	Yes	No

- Model 2: Performs the best -with biggest positive impact of adding regularization & early stopping
- Model 3 & 4: Shows slightly lower performance than 2, showing that increasing layers & hyperparameter tuning alone does not give the best results
- Key Learnings: To combine regularization with hyperparameter tuning in future to further improve results

Future work to improve classification

Please identify me better next time!

- Challenges:
  - Current validation accuracy is still not ideal.
- Next Possible Steps:
  - Combine Hyperparameter Tuning with regularization and early stopping.
- Increase Dataset Size:
  - Use GANs to generate more training samples.
- Goal: Achieve higher validation accuracy and improved model generalization.



Thank you!

Github Link