Decreasing Learning Rate vs Increasing Batch Size in Image Classification

Galen Yacalis

Quan Le

Overview

Topic and approach

Evaluating the differences between increasing batch size and decreasing learning rate during training.

Create a model that, when given a headshot, predicts whether the person is wearing glasses, whether the person is smiling, and what their gender is.

Training the network

We trained our neural network in 3 ways

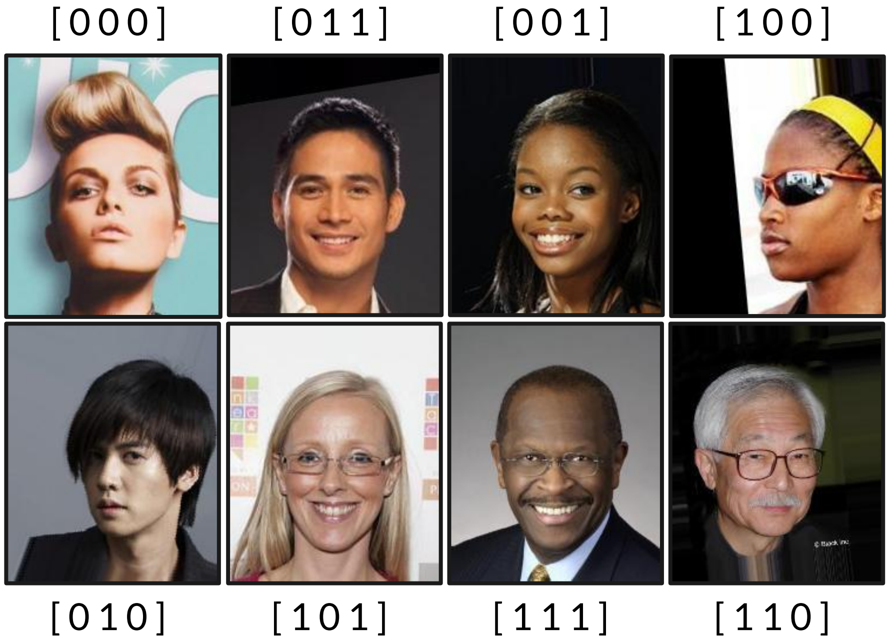
1. The batch size and learning rate stay constant
2. Increase the batch size by α when the validation loss starts to plateau
3. Decrease the learning rate by α when the validation loss starts to plateau

Dataset

We tried two different datasets, each of 100k+ images of celebrities.

The first dataset was IMDB and included age and gender labels. But, we discovered it was about 20% mislabeled and ended up throwing it out.

The second dataset was CelebA. We used 3 binary labels for multi-label classification: Eyeglasses, Male, and Smiling.



Dataset

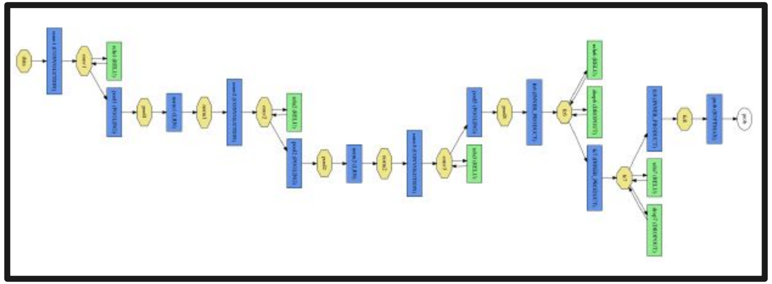
CelebA preprocessing:

1. Remove all images labeled as blurry
2. Resize to 178 x 218
3. Change label values from (-1, 1) to (0, 1)

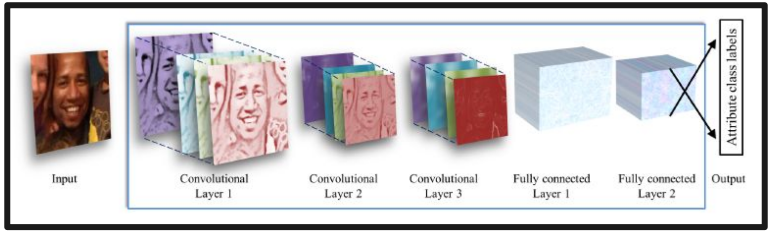
[0 0 0] [0 1 1] [0 0 1] [1 0 0]

[0 1 0] [1 0 1] [1 1 1] [1 1 0]

Very large networks performed poorly



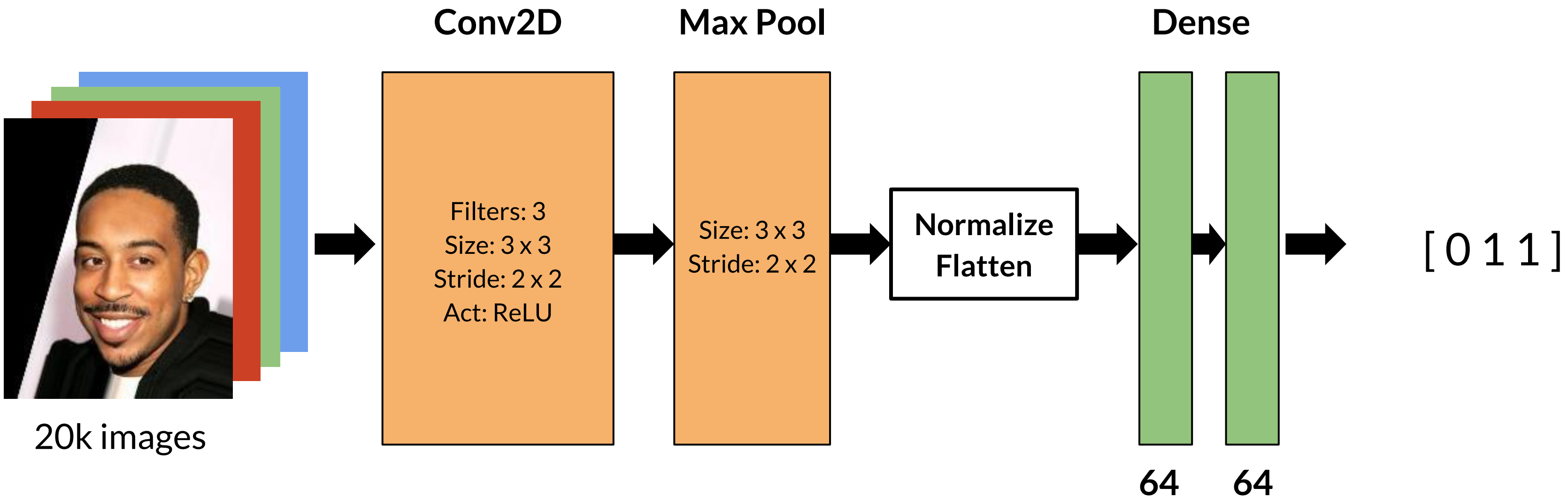
~ 56 %



~ 56 %

~ 93 %

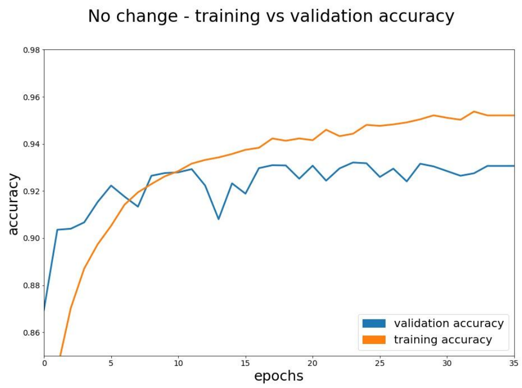
Final CNN



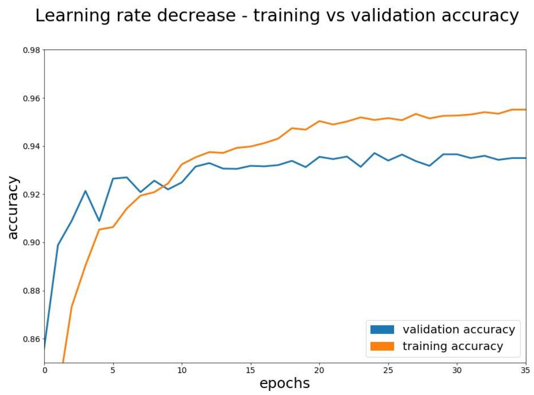
Total parameters: 460,707

Accuracy vs epoch

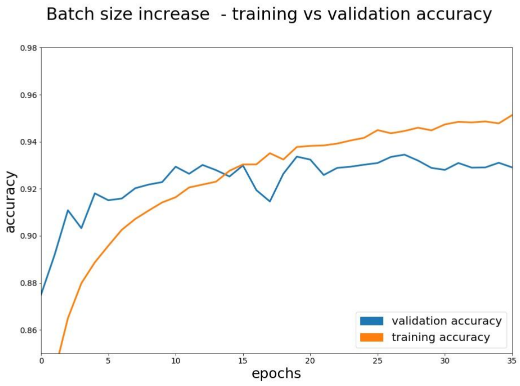
No Change



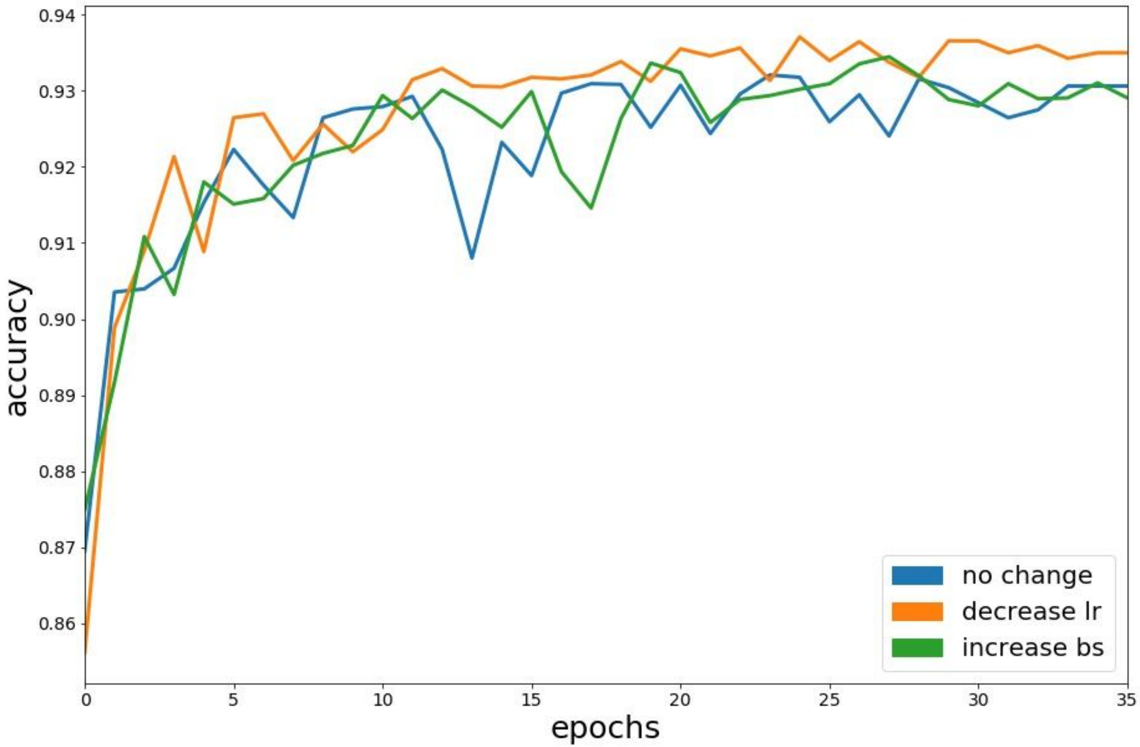
Changing the Learning Rate



Changing the Batch size



Accuracy vs epoch



Accuracy comparison

Data	No change	Learning rate decrease	Batch size increase
Training Data	0.952	0.955	0.951
Validation Data	0.931	0.935	0.930
Test Data	0.933	0.936	0.931

Classification report for best model

Class	Precision	Recall	F1 Score	Support
Glasses	0.95	0.84	0.89	74
Male	0.90	0.97	0.93	539
Smiling	0.90	0.86	0.88	487
Avg / Total	0.90	0.91	0.91	1100

Examples

All correct

81 %

One or two wrong

19 %

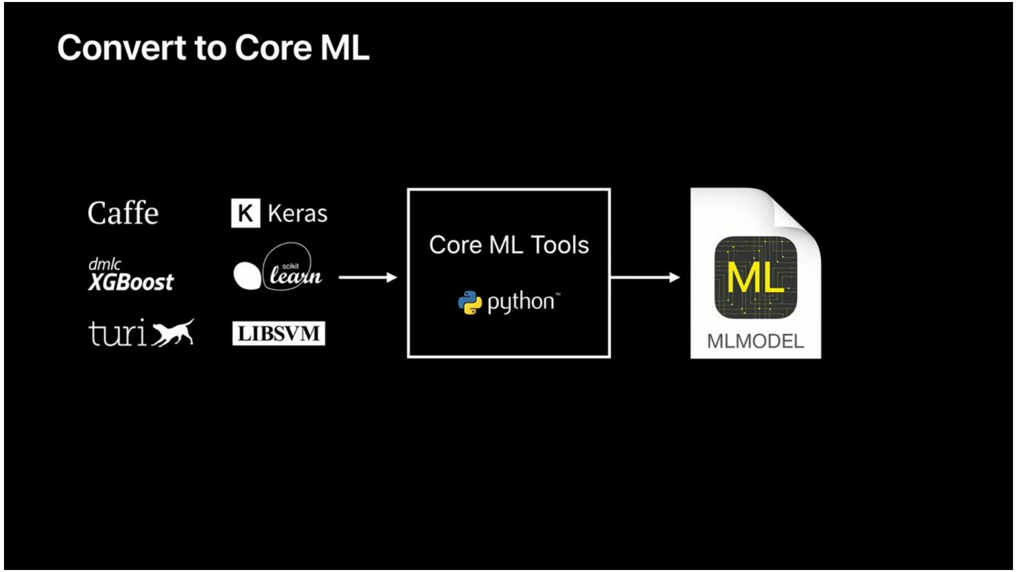
All wrong

< 1 %

Using Apple's Core ML

Can turn keras or other custom model into a coreml model, and integrate it into an app.

Sadly, image resizing is obscured by API and skews results.



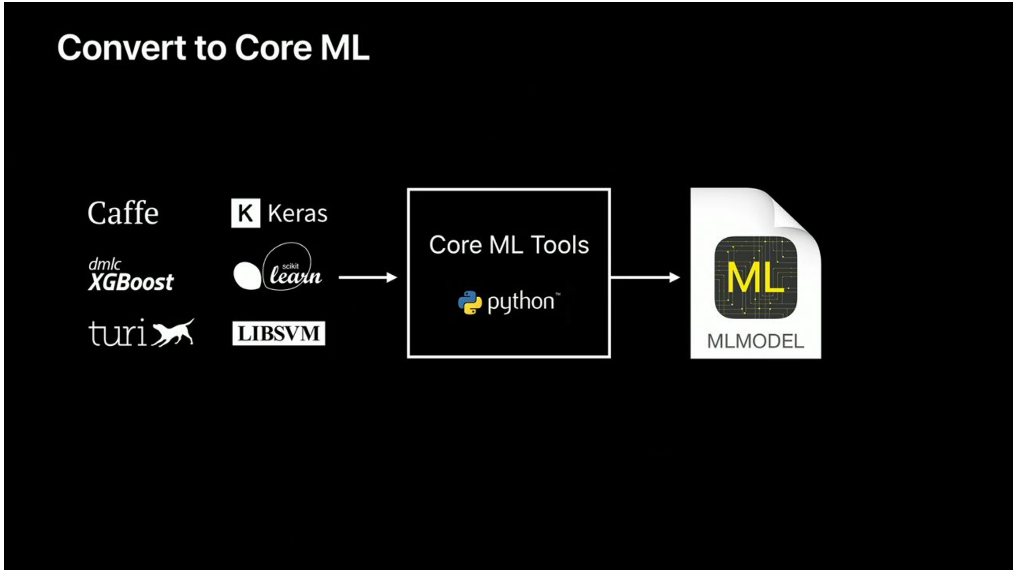
Using our keras model in an iOS app

Steps for image classification:

- Use coreml tools to convert keras model to coreml and import into app
- Determine how image will be chosen
- Within app, request an asynchronous model prediction on the image
- Update the text on the screen with the results of the prediction

Advantages of native Swift code and coreML:

- API is optimized for iOS GPU
- Swift 4 code works fast on hardware



Future Directions

In our case, increasing the batch size did not improve performance.

We could try different models, or different parameters.

We received many suggestions that different initialization values would make a huge difference - this is the first thing that should be tried.

Additionally, an optimizer other than Adam might work better for getting differences between batch size and learning rate.

The app needs to work better in resizing images - it does it automatically now, but not very well.

[0 1 1] [0 1 0] [0 0 0]

References

Levi, G., & Hassner, T. (2015). Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 34-42).

Devarakonda, A., Naumov, M., & Garland, M. (2017). AdaBatch: Adaptive Batch Sizes for Training Deep Neural Networks. *arXiv preprint arXiv:1712.02029*.

Smith, S. L., Kindermans, P. J., & Le, Q. V. (2017). Don't Decay the Learning Rate, Increase the Batch Size. *arXiv preprint arXiv:1711.00489*.

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 3730-3738).

Thank you