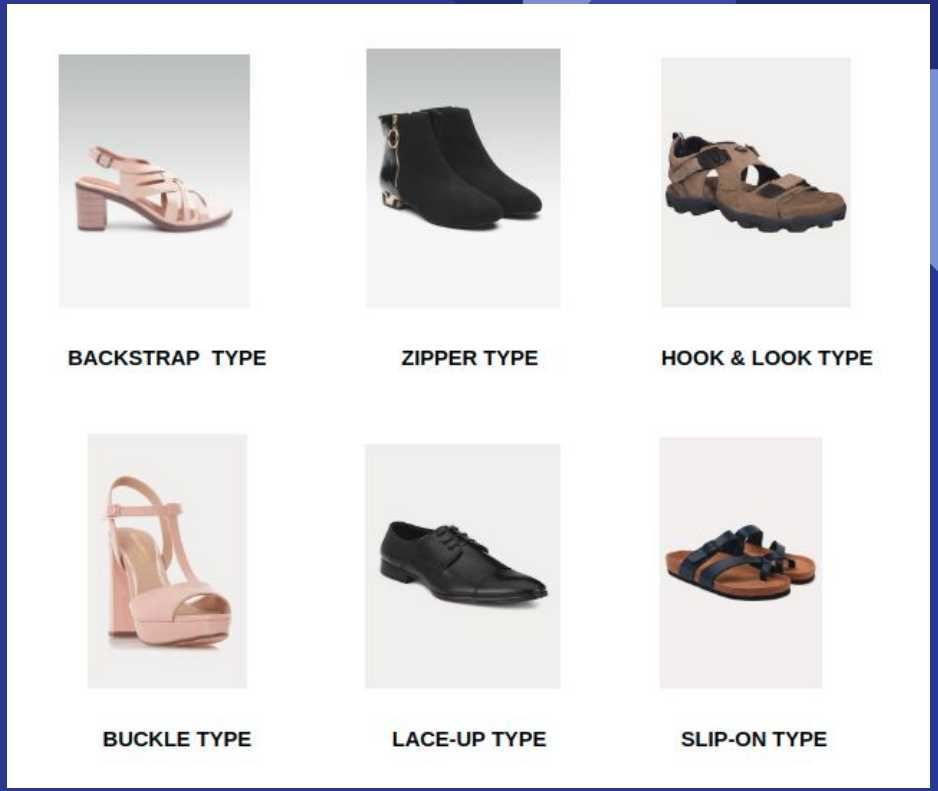
Fynd

Machine Learning Challenge Solution

the solution to Fynd’s problem



Contents

1. The Problem

Closure type classification of 6 different types of footwears:

i. Back Strap Closure
ii. Buckle Closure
iii. Hook & Look Closure
iv. Lace-up Closure
v. Slip On Closure
vi. Zip Closure

2. The Approach

Data:

- We have used only the ***view-1*** link to download the data.
- We divide the provided data with train + validation sets (80:20)

Training:

We have got a fine-grained classification problem with 6 classes.One of the crucial problem in training NN is to find a good learning rate for your data. We will address this issue with the followings:

- We will be using Cyclical Learning Rates for finding optimal learning rate for our data
- Then we will use One Cycle Policy to train our model with the found optimal learning rate(s).

N.B: Recently I have written a blog post on the 1 cycle policy, which can be found here.

Training setup

Model: *Resnet50*
Loss Function: *Cross Entropy*
Optimizer: *SGD with Momentum*

- We first freeze our model and use learning rate range test to find an optimal learning rate for our data.
- Next we train our freezed model for some epochs with 1 cycle policy
- Then we unfreeze our model and and again find optimal lr.
- Again we train our unfreeze model with 1 cycle policy for more no. of epochs

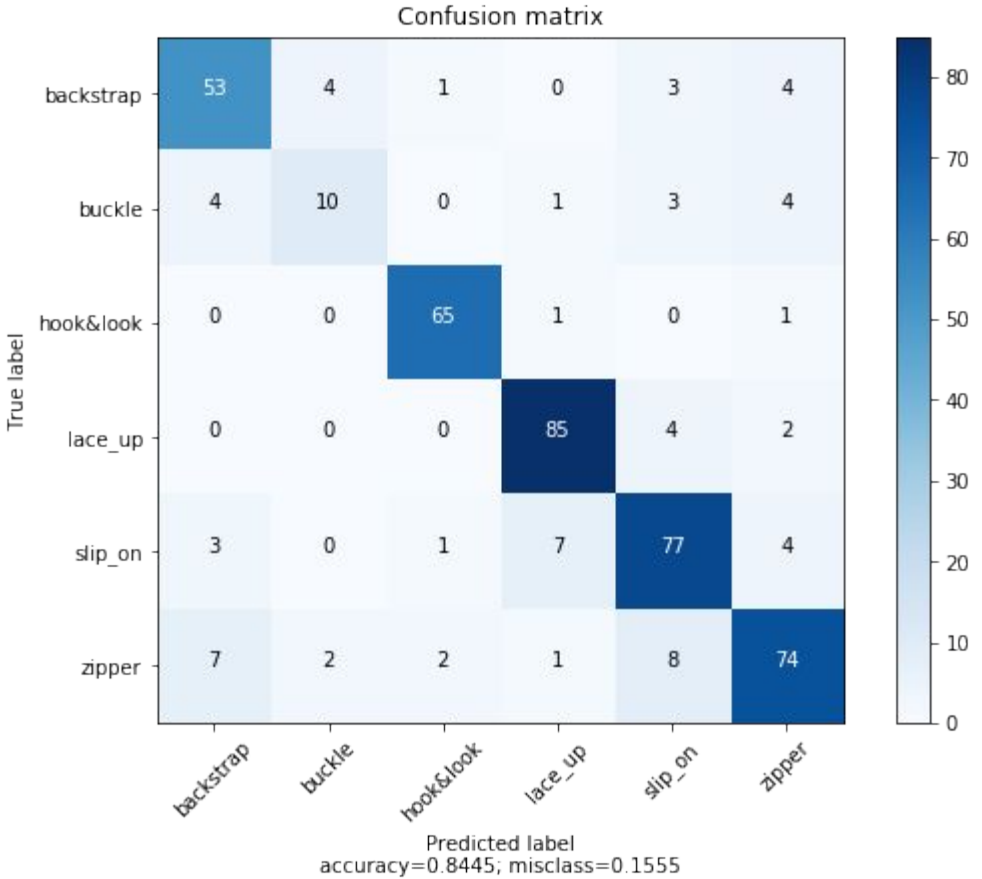
After nearly 25 epochs, we are getting ~80% accuracy on the validation set.

9	0.38298	0.69061	80.00000	80.00000
10	0.36808	0.72445	81.00000	80.00000

3. The Results

Interpretations:

Our model have correctly classified 364 out of 431 images i.e. achieved an accuracy of 84.45%.
Here is the Confusion matrix:



4. Room for improvements

Q1. Need more data?

A: Yes. At some point of observation the accuracy will not improve and is stuck, which is a strong indication for the need of more data.

2. The Data collection:

I. Google Image scraping.

II. Look for similar datasets. (e.g. search in kaggle.com)

III. Use specific API to collect data from merchants (e.g. amazon.com API)

Hyperparameters used:

1. **Learning Rate:** found from LR finder (2e-3, 2e-4)
2. **Momentum Values:** (0.95, 0.85) as mentioned in 1 cycle policy paper.
3. **Weight Decay:** 1e-4 as directed in the same paper.
4. **Batch Size:** 32 (as it's the highest number that fitted into my gpu's memory)

Other Questions:

Q2. Key insights about the public dataset given:

- Good yet imbalanced dataset.
- Feel like some images are not placed in correct class (insight from manual inspection). [Not sure about this point]

Q3. What else can you do for better performance?

- More Data Augmentation
- More Data
- Removing Noise in Data
- Cosine Annealing of learning rates instead of Linear
- Using some latest architectures like Efficient-Net

References:

- One Cycle Policy: https://github.com/nachiket273/One_Cycle_Policy
- Download Dataset: https://www.kaggle.com/abinesh100/easy-download-images-in-25-lines-py3/code
- Confusion Matrix: https://www.kaggle.com/grfiv4/plot-a-confusion-matrix
- Blog Post: https://medium.com/dsnet/the-1-cycle-policy-an-experiment-that-vanished-the-struggle-in-training-neural-nets-184417de23b9

Presented By:

Shubhajit Das
ML Enthusiast