ACSE-4.4 Machine Learning Project

Team Divergence

Contents

1. Teamwork organization
2. Data Processing
3. Experiments
4. Results
5. Conclusion
6. Appendix

Teamwork organization

- Daily Catch-Up meeting
- TA meeting sessions
- Microsoft Team posts
- Github/Google Drive sharing

Data processing

- Store as numpy array
- Use random shuffle split
- Balanced train and validate set
- Normalisation, value from ImageNet

Experiments

(* - used for submission)

- Hand-crafted networks (Mon)
  - LeNet-5*
  - VGG*
  - GoogleNet
- Pytorch pretrained models (Tue - Thu)
  - VGG*
  - GoogleNet
  - ResNet (18*/34/50 layers)
  - Wide ResNet* (50 layers)
- Ensemble methods (Wed - Thu)
  - Majority voting*
  - Per-class weighted average*
  - Boost

Fine tuning our models

- Baseline
- Hyperparameters (Weight decay, Training-validation split ratio, Learning rate, batch size, test batch size, random seed)
- Data augmentation techniques
- Whether or not to train on the full training set
- Whether or not we retrain the entire model or only the last layer (ResNet-18/GoogleNet)

Data Augmentation

Random Horizontal Flip

Random Rotation up to 10 degree

Add Gaussian noise

(take standard deviation as hyperparameters, candidate values include 0.002, 0.004 and 0.005)

Hand-crafted Models

+ Naturally follows assignment 8.2
+ Easy to implement and tune
+ Less training time

- Weak performance

< 40%

Pytorch Pretrained Models

VGG-16

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	16
Test Batch Size	1000

62%

Pytorch Pretrained Models

GoogleNet

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	128
Test Batch Size	1000

65.5%

Pytorch Pretrained Models

ResNet-18

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	64
Test Batch Size	1000

66%

Pytorch Pretrained Models

ResNet-34

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	128
Test Batch Size	1000

71.1%

Pytorch Pretrained Models

ResNet-50

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	128
Test Batch Size	1000

74.4%

Pytorch Pretrained Models

Wide ResNet-50-2

Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	64/128/256
Test Batch Size	1000

79-80%

Ensemble methods

Majority Voting
Make predictions on the test dataset with every single model in the ensemble, and determine the final prediction using majority voting

- No differentiation of strong/weak models
- Biased results when using multiple versions of one model

Ensemble methods

Boost
Similar to majority voting, but assigning different weights to different models in the ensemble. These weights are learned.

- + Could take advantage of both weak and strong models

Ensemble methods

Per-class weighted averaging
Instead of averaging the final predictions, perform the weighted averaging on the likelihood of each individual class provided by all the models

+ Could take advantage of both weak and strong models

Scoreboard

Model	Validation set accuracy	F1 score private	F1 score public
Handcrafted LeNet	0.20	0.09952	0.09724
Handcrafted VGG	0.40	0.39649	0.40282
VGG-16	0.62	0.62588	0.62719
GoogleNet	0.655		
ResNet-18	0.66	0.66029	0.66023
ResNet-34	0.711		
ResNet-50	0.744		
Wide ResNet-50-2	0.80	0.79205	0.79601
Majority Voting		0.79446	0.79334
Boost	0.929	0.75441	0.75763
Per-class average	0.887	0.78418	0.78924

Results

Ranked 7 in the private leaderboard

Using the Wide ResNet-50-2, trained on the 90% training set

F1 score: 0.79601 on the public subset, 0.79205 on the private subset

Conclusion

Transferred learning is the key approach

CUDA out of memory

Avoid over-fitting

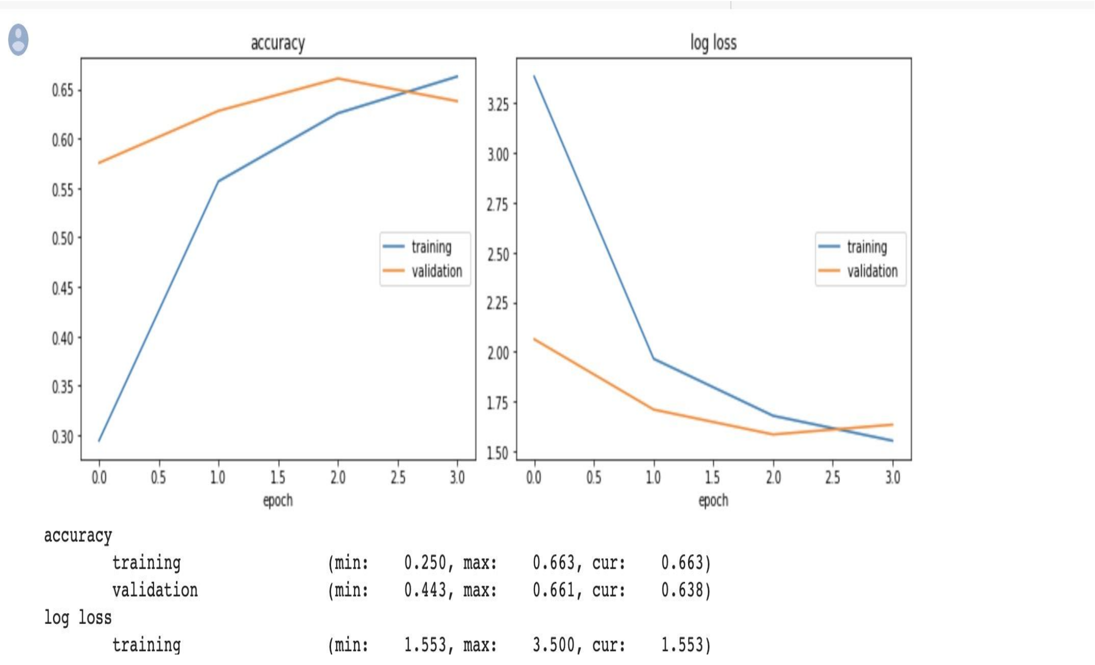
Ensemble method is more suitable for 0-1 classification, not for 200 classes

WideResNets50

	Train Set Accuracy	Validation Set Accuracy	Train Set Log Loss	Validation Set Log Loss	Full Data Set Accuracy
Version 1	0.857	0.789	0.730	0.926	
Version 2	0.810	0.807	0.968	1.019	
Version 3	0.875	0.793	0.692	0.930	0.824

	Momentum	Weight Decay	Validation Set Split Ratio	Learning Rate	Batch Size	Test Batch Size
Version 1	0.5	1e-2	0.1	1e-2		
Version 2	0.5	1e-2	0.2	1e-2	256	1000
Version 3	0.5	1e-2	0.2	1e-2	128	1000

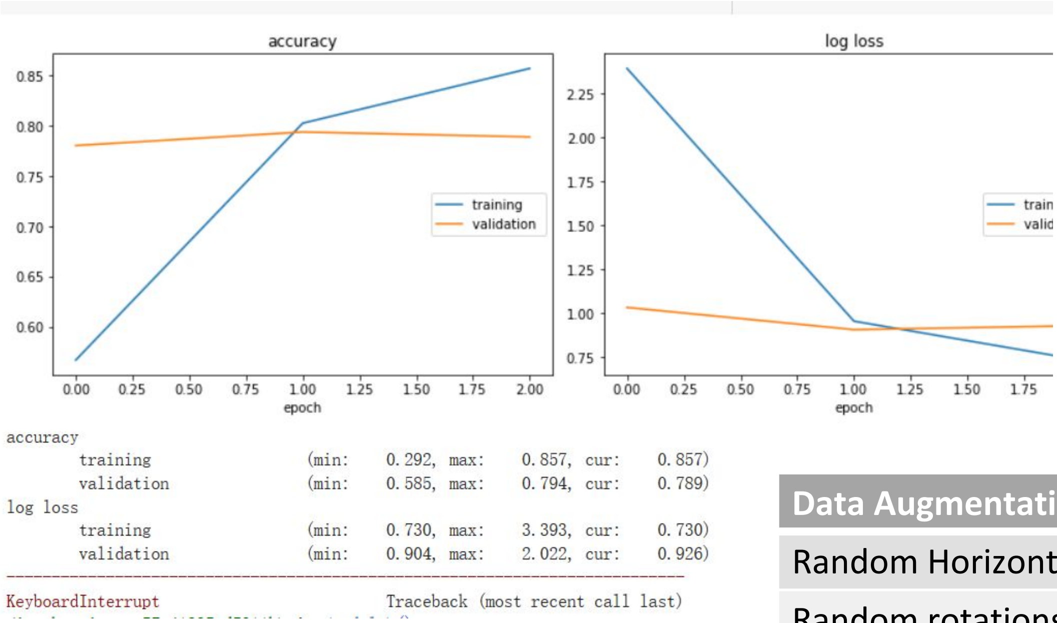
ResNet18



```
accuracy
	training           	 (min:    0.250, max:    0.663, cur:    0.663)
	validation         	 (min:    0.443, max:    0.661, cur:    0.638)
log loss
	training           	 (min:    1.553, max:    3.500, cur:    1.553)
```

Momentum	0.5
Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	64
Test Batch Size	1000

Version 1

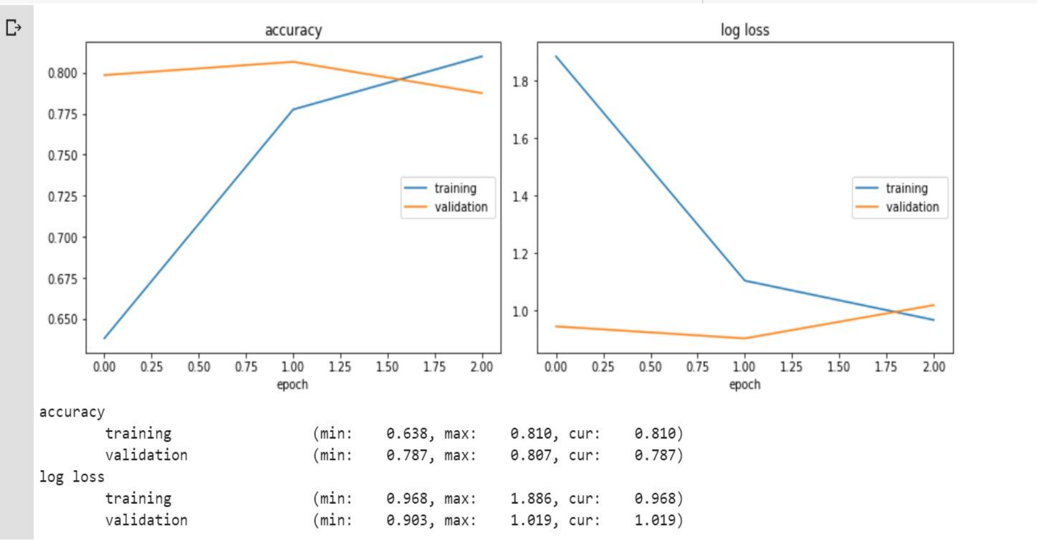


```
accuracy
        training             (min:    0.292,  max:    0.857,  cur:    0.857)
        validation           (min:    0.585,  max:    0.794,  cur:    0.789)
log loss
        training             (min:    0.730,  max:    3.393,  cur:    0.730)
        validation           (min:    0.904,  max:    2.022,  cur:    0.926)
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
```

Momentum	0.5
Weight Decay	1e-2
Validation Set Split Ratio	0.1
Learning Rate	1e-2
Batch Size	
Test Batch Size	

Data Augmentation
Random Horizontal Flip
Random rotations between (0, 10)

Version 2

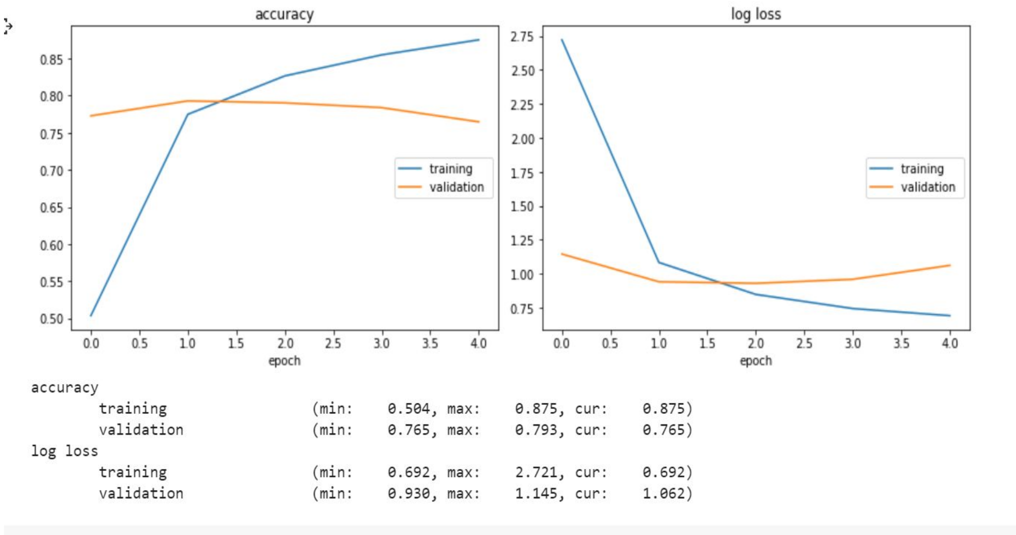


```
accuracy
	training           	 (min:    0.638, max:    0.810, cur:    0.810)
	validation         	 (min:    0.787, max:    0.807, cur:    0.787)
log loss
	training           	 (min:    0.968, max:    1.886, cur:    0.968)
	validation         	 (min:    0.903, max:    1.019, cur:    1.019)
```

Momentum	0.5
Weight Decay	1e-2
Validation Set Split Ratio	0.2
Learning Rate	1e-2
Batch Size	256
Test Batch Size	1000

Data Augmentation
Random Horizontal Flip
Random Rotation between (0, 10)

Version 3

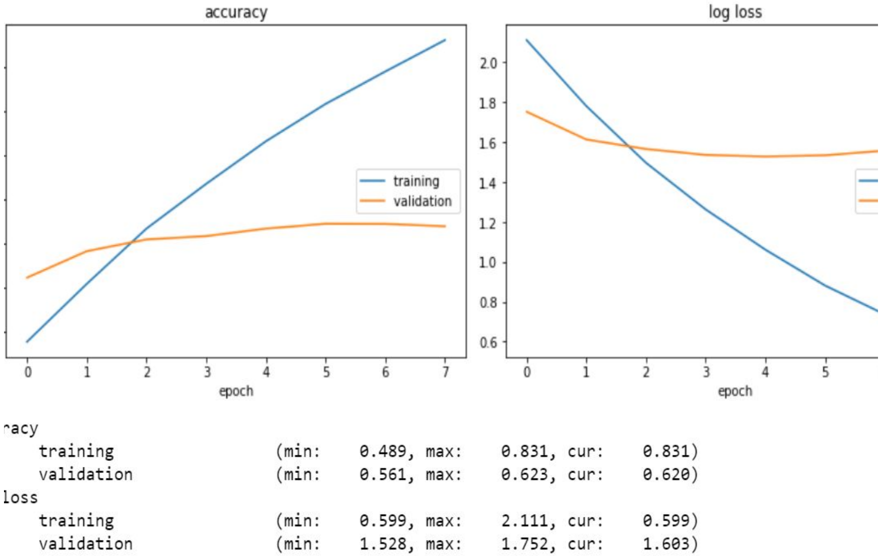


```
accuracy
	training           	 (min:    0.504, max:    0.875, cur:    0.875)
	validation         	 (min:    0.765, max:    0.793, cur:    0.765)
log loss
	training           	 (min:    0.692, max:    2.721, cur:    0.692)
	validation         	 (min:    0.930, max:    1.145, cur:    1.062)
```

Momentum	**0.5**
Weight Decay	1e-2
Validation Set Split Ratio	0.2
Learning Rate	1e-2
Batch Size	128
Test Batch Size	1000

Data Augmentation
Random Horizontal Flip
Random Rotation between (0, 10)
Add Gaussian Noise

VGG16



```
racy
    training            (min:    0.489, max:    0.831, cur:    0.831)
    validation          (min:    0.561, max:    0.623, cur:    0.620)
loss
    training            (min:    0.599, max:    2.111, cur:    0.599)
    validation          (min:    1.528, max:    1.752, cur:    1.603)
```

Momentum	**0.5**
Weight Decay	1e-2
Validation Set Split Ratio	0.2
Learning Rate	1e-2
Batch Size	16
Test Batch Size	100

Data Augmentation
Random Horizontal Flip
Random Rotation between (0, 10)

Restrictions

1. Memory Limitations of CUDA, JupyterHub etc.

   (Deeper models are always winners)
2. Balance between High Accuracy and Running Time

   (learning rate, momentum etc.)
3. Different Time Zones and Short Duration of Project

4. Only 2 Submissions per day

Results

1. Ranked 7 in the private leaderboard
2. Using the Wide ResNet-50-2, trained on the 90% training set
3. F1 score: 0.79601 on the public subset, 0.79205 on the private subset